1700
1700
1700
1700

MSOS

CONTROL DATA
CORPORATION

INSTALLATION HANDBOOK

1700

1700

1700

1700

MSOS

CONTROL DATA
CORPORATION

INSTALLATION HANDBOOK

REVISION RECORD

REVISION	NOTES
A	Add sections Part I, 5-1 and Part II, 5-1
(8-68)	
B	This manual obsoletes the 1700 Mass Storage Operating System (MSOS) Product Set Installation
(1-70)	Handbook Revision A.
C	Add Macro Assembler 2.0, 1726-405 card reader driver, 1732-608/609 magnetic tape driver,
(3-70)	and 1740-501 line printer driver release information as well as corrections.
D	Add System Checkout and System Configurator as well as corrections.
(6-70)	
E	Add 1777 paper tape station as well as corrections.
(8-70)	

Additional copies of this manual may be obtained from the nearest Control Data Corporation sales office.

Pub. No. 60234300

Address comments concerning this manual to:

Control Data Corporation
Software Documentation
4201 North Lexington Avenue
St. Paul, Minnesota 55112

or use Comment Sheet in the back of this manual.

CONTENTS

PART I

RELEASE SUMMARY

PART II

INSTALLATION PROCEDURES

PART III

INSTALLATION RELATED INFORMATION

PART I

RELEASE SUMMARIES

1700 MSOS 2.1 RELEASE SUMMARY 1

1.1 PRODUCTS

Version 2.1 of the Mass Storage Operating System (MSOS) is accompanied by the following product set members:

1700 Macro Assembler 2.0

1700 Mass Storage FORTRAN 2.0A

1700 Mass Storage FORTRAN 2.0B

1700 COSY 1.0

1700 System Checkout

1700 System Configurator

1.2 RELEASE MATERIALS

Materials issued to the user with the system, as well as optional materials available to the user on request, are listed below.

1.2.1 MSOS 2.1

Paper Tape Version

One system initializer paper tape

Six installation paper tapes

Magnetic Tape Version

One system initializer paper tape

One installation magnetic tape

Optional Tapes

One COSY source magnetic tape

Two list magnetic tapes

1.2.2 MACRO ASSEMBLER 2.0

Paper Tape Version

One installation paper tape containing relocatable programs and control cards

One installation verification deck

Magnetic Tape Version

One installation magnetic tape containing relocatable programs and control cards

One installation verification deck

Optional Tapes

One installation paper tape in relocatable format containing library macro preparation programs

One installation paper tape in ASCII format containing system library macros

One COSY source magnetic tape

One list magnetic tape

1.2.3 MASS STORAGE FORTRAN 2.0A

Paper Tape Version

One paper tape containing SELCOP and IOCAL

Sixteen installation paper tapes

One installation verification program

Magnetic Tape Version

One paper tape containing SELCOP and IOCAL

One installation magnetic tape

One installation verification program

Optional Tapes

One COSY source magnetic tape

Three list magnetic tapes

1.2.4 MASS STORAGE FORTRAN 2.0B

Paper Tape Version

One paper tape containing SELCOP and IOCAL

Ten installation paper tapes

One installation verification program

Magnetic Tape Version

One paper tape containing SELCOP and IOCAL

One installation magnetic tape

One installation verification program

Optional Tapes

One COSY source magnetic tape

Three list magnetic tapes

1.2.5 COSY 1.0

Paper Tape Version

One paper tape

One installation verification deck

Magnetic Tape Version

One magnetic tape

One installation verification program

Optional Tapes

One COSY source magnetic tape

1.2.6 1745-2 DISPLAY DRIVER

Buffered Version

One installation paper tape

Unbuffered Version

One installation paper tape

Optional Tapes

One paper tape buffered source

One paper tape unbuffered source

One magnetic tape buffered and unbuffered source

1.2.7 1713 TELETYPEWRITER

Paper Tape Version

Four paper tapes

Magnetic Tape Version

One magnetic tape

Optional Tapes

One COSY source magentic tape

One list magnetic tape

1.2.8 1726-405 CARD READER DRIVER

Paper Tape Version

One paper tape of driver in relocatable binary

Magnetic Tape Version

One magnetic tape of driver in relocatable binary

Optional Tapes

One COSY source magnetic tape

1.2.9 1732-608/609 MAGNETIC TAPE DRIVER

Paper Tape Version

Two paper tapes of driver modules in relocatable binary

Magnetic Tape Version

One magnetic tape of driver modules in relocatable binary

Optional Tapes

One COSY source magnetic tape

1.2.10 1740-501 LINE PRINTER DRIVER

Paper Tape Version

One paper tape of driver in relocatable binary

Magnetic Tape Version

One magnetic tape of driver in relocatable binary

Optional Tapes

One COSY source magnetic tape

1.2.11 SYSTEM CHECKOUT

Paper Tape Version

Six installation tapes

Magnetic Tape Version

One installation tape

Optional Tapes

One COSY source magnetic tape

One list magnetic tape

1.2.12 SYSTEM CONFIGURATOR

Paper Tape Version

Twelve binary installation tapes

Fourteen binary paper tapes containing definitions and skeletons

Magnetic Tape Version

One installation tape

One tape containing system definitions and skeletons

Optional Tapes

One COSY source magnetic tape

One list magnetic tape

1.2.13 1777 PAPER TAPE STATION

One COSY magnetic tape

One relocatable binary magnetic tape containing paper tape images

One hollerith magnetic tape containing source program from which cards can be punched

One list magnetic tape

1.3 TAPE STRUCTURES

For further explanation of tape content, see Part II, Section 4.3.

1.3.1 MSOS 2.1 SYSTEM INITIALIZER AND INSTALLATION PAPER TAPES

System Initializer

ABSOLUTE FORMAT RECORD 1 CHECKSUM LOADER
RECORD 2 SYSTEM INITIALIZER

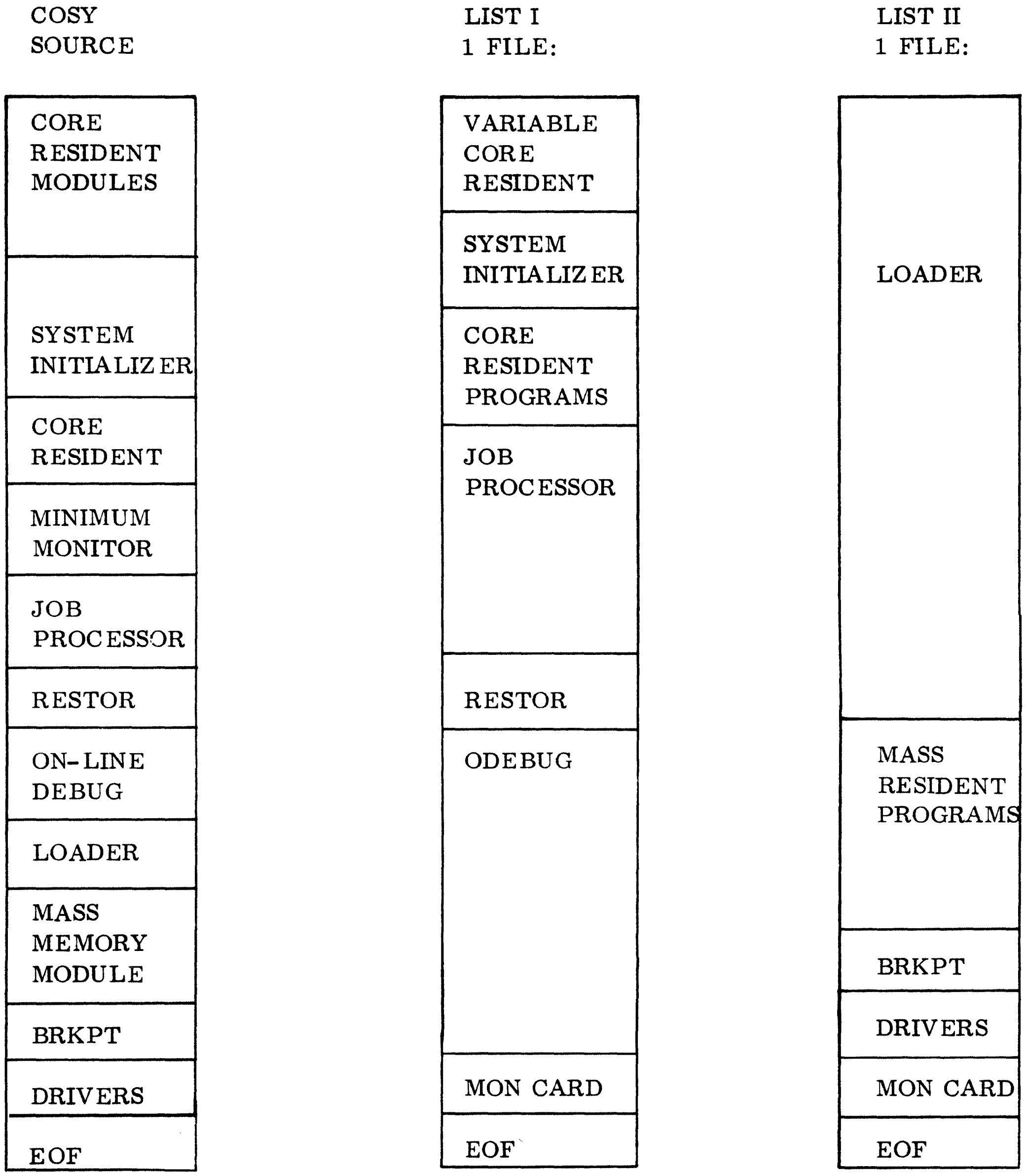
Installation Paper Tapes

PAPER TAPE 1	PAPER TAPE 2	PAPER TAPE 3	PAPER TAPE 4	PAPER TAPE 5	PAPER TAPE 6
CORE RESIDENT MODULE	LOADER	REMAINING CORE RESIDENT MODULES	JOB PROCESSOR	RESTOR	DRIVERS
		JOB PROCESSOR	LIBEDT	ODEBUG	REQUEST PRIORITY ASSIGNMENTS
				BRKPT	

1.3.2 MSOS 2.1 MAGNETIC INSTALLATION TAPE

SYSBUF
STANDARD SYSTEM
LOADER
CORE RESIDENT MODULES
JOB PROCESSOR
LIBEDT
RESTOR
ODEBUG
BRKPT
DRIVERS
REQUEST PRIORITY ASSIGNMENTS
EOF

1.3.3 MSOS 2.1 OPTIONAL SOURCE AND LIST MAGNETIC TAPES



COSY SOURCE

- CORE RESIDENT MODULES
- SYSTEM INITIALIZER
- CORE RESIDENT
- MINIMUM MONITOR
- JOB PROCESSOR
- RESTOR
- ON-LINE DEBUG
- LOADER
- MASS MEMORY MODULE
- BRKPT
- DRIVERS
- EOF

LIST I 1 FILE:

- VARIABLE CORE RESIDENT
- SYSTEM INITIALIZER
- CORE RESIDENT PROGRAMS
- JOB PROCESSOR
- RESTOR
- ODEBUG
- MON CARD
- EOF

LIST II 1 FILE:

- LOADER
- MASS RESIDENT PROGRAMS
- BRKPT
- DRIVERS
- MON CARD
- EOF

1.3.4 MACRO ASSEMBLER 2.0 PAPER AND MAGNETIC TAPES

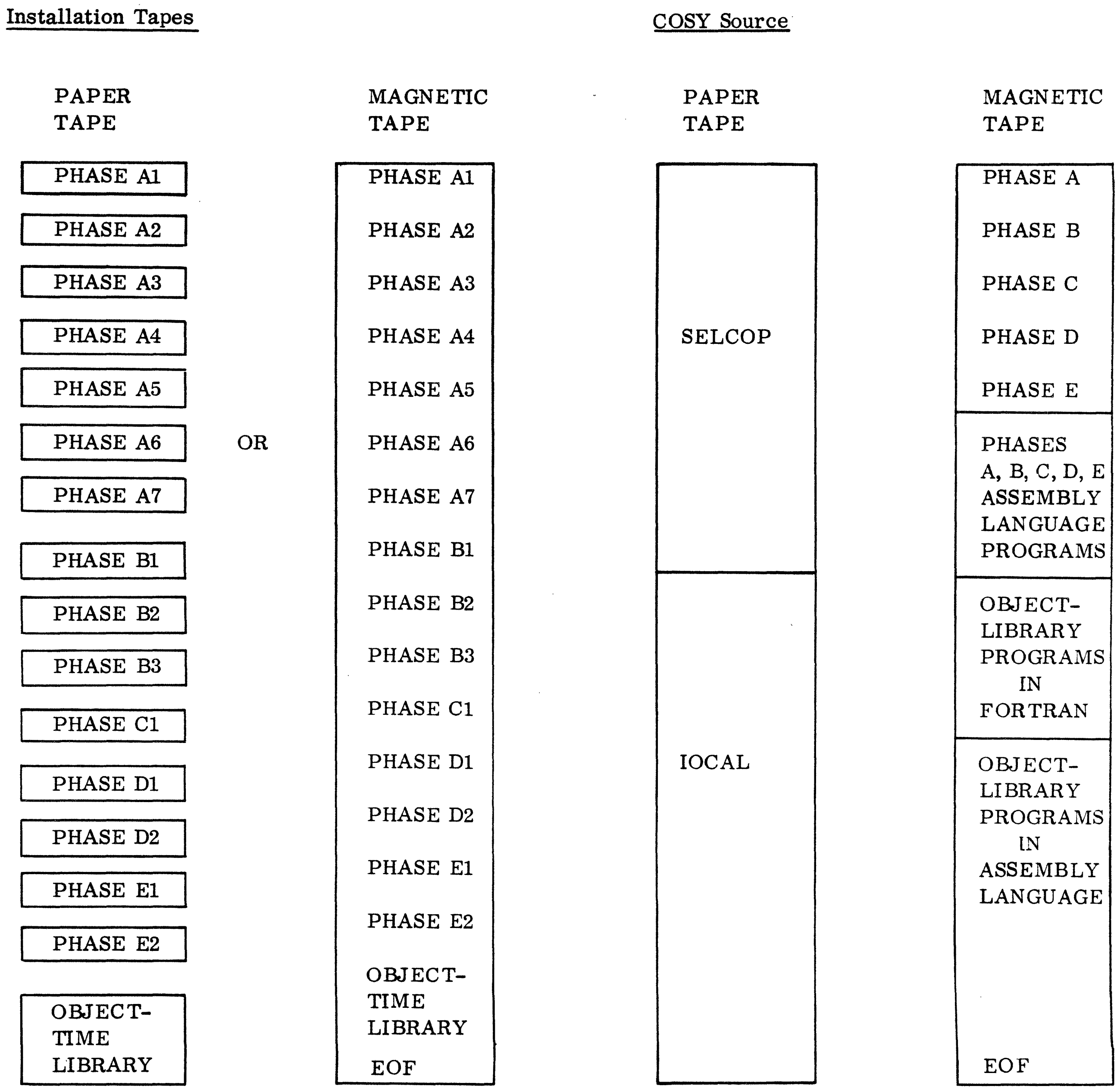
Installation Tapes			Optional Tapes	
TAPE 1 PAPER TAPE		TAPE 4 MAGNETIC TAPE	TAPE 2 PAPER TAPE	TAPE 3 PAPER TAPE
INSTALLATION FROM PAPER TAPE IN RELOCATABLE BINARY INCLUDES CONTROL STATEMENTS AND ASSEM PASS1 PASS2 PASS3 PASS4 AND ABSOLUTIZED MACSKL AND MACROS	OR	INSTALLATION FROM MAGNETIC TAPE IN RELOCATABLE BINARY INCLUDES CONTROL STATEMENTS AND ASSEM PASS1 PASS2 PASS3 PASS4 AND ABSOLUTIZED MACSKL AND MACROS EOF	LIBRARY MACRO PREPARATION PROGRAM RELOCATABLE BINARY OF LIBMAC LIBMC2 LIBMC3	SYSTEM LIBRARY MACROS IN ASCII SOURCE

Optional Tapes

SOURCE TAPE 5	LIST TAPE 6
COSY SOURCE OF:	LIST
ASSEM PASS1 PA1 PR2 PASS2 PA2 PR2 PASS3 PA3 PR2 PA3 PR3 PASS4	
LIBMAC LIBMC2 LIBMC3	
EOF	EOF

1.3.5 MASS STORAGE FORTRAN 2.0A PAPER AND MAGNETIC TAPES

The installation material is on one magnetic tape or on 16 paper tapes with each phase on a different paper tape. Source information is on one magnetic tape in COSY format. SELCOP and IOCAL are on one paper tape. There are three list magnetic tapes.



Installation Tapes

PAPER TAPE

PHASE A1

PHASE A2

PHASE A3

PHASE A4

PHASE A5

PHASE A6

PHASE A7

PHASE B1

PHASE B2

PHASE B3

PHASE C1

PHASE D1

PHASE D2

PHASE E1

PHASE E2

OBJECT-TIME LIBRARY

OR

MAGNETIC TAPE

PHASE A1

PHASE A2

PHASE A3

PHASE A4

PHASE A5

PHASE A6

PHASE A7

PHASE B1

PHASE B2

PHASE B3

PHASE C1

PHASE D1

PHASE D2

PHASE E1

PHASE E2

OBJECT-TIME LIBRARY

EOF

COSY Source

PAPER TAPE

SELCOP

IOCAL

MAGNETIC TAPE

PHASE A

PHASE B

PHASE C

PHASE D

PHASE E

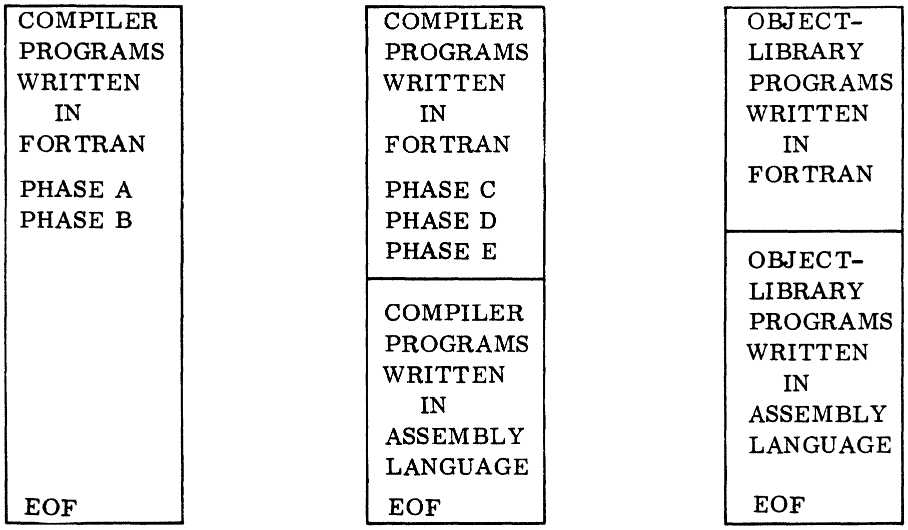
PHASES A, B, C, D, E ASSEMBLY LANGUAGE PROGRAMS

OBJECT-LIBRARY PROGRAMS IN FORTRAN

OBJECT-LIBRARY PROGRAMS IN ASSEMBLY LANGUAGE

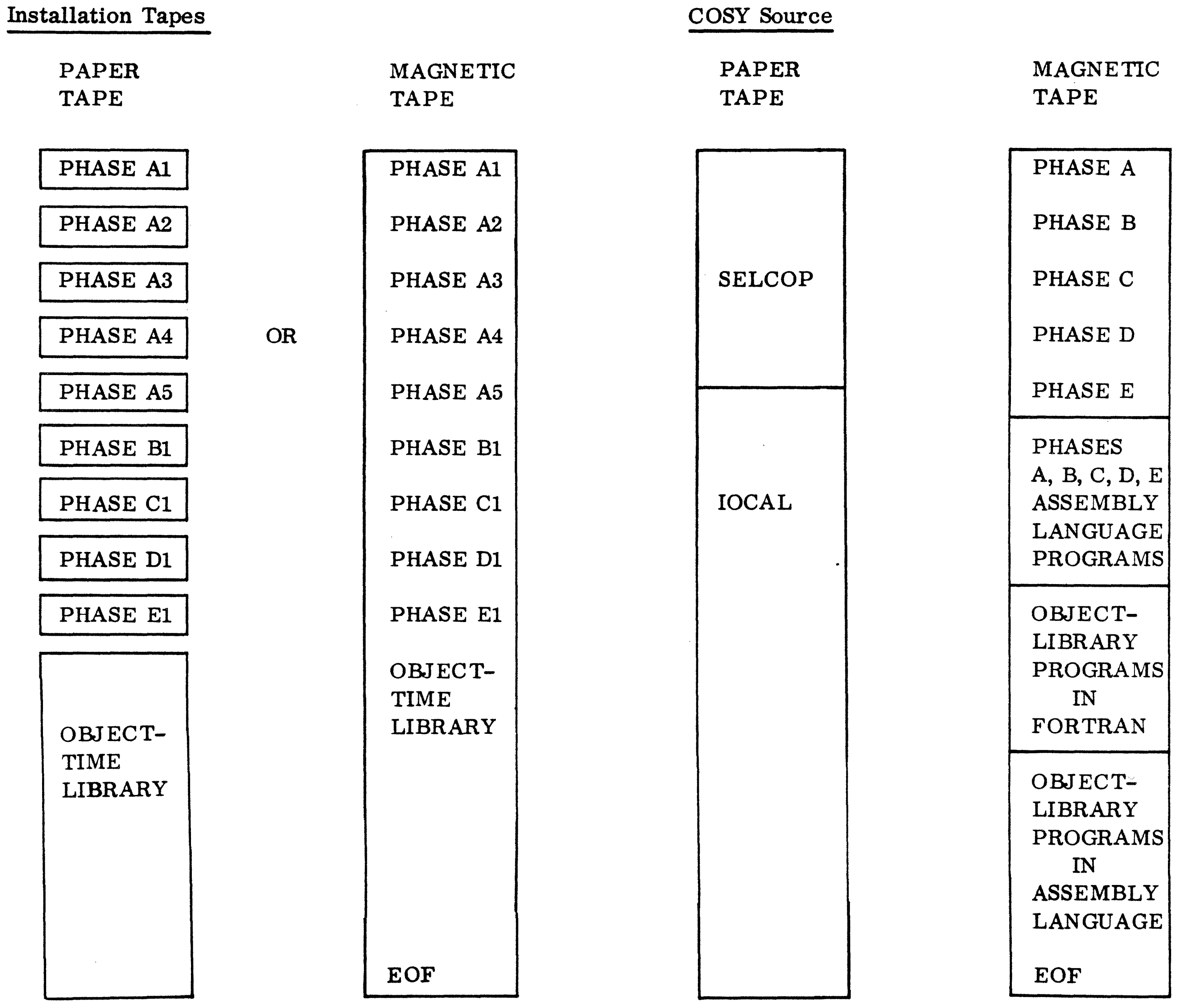
EOF

List Magnetic Tapes for 2.0A

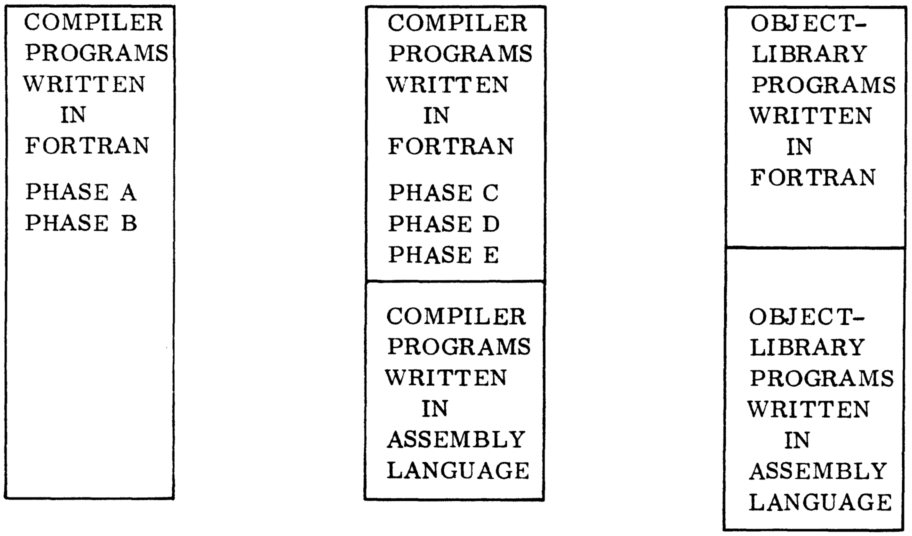

COMPILER PROGRAMS WRITTEN IN FORTRAN
PHASE A
PHASE B
EOF
COMPILER PROGRAMS WRITTEN IN FORTRAN
PHASE C
PHASE D
PHASE E
COMPILER PROGRAMS WRITTEN IN ASSEMBLY LANGUAGE
EOF
OBJECT-LIBRARY PROGRAMS WRITTEN IN FORTRAN
OBJECT-LIBRARY PROGRAMS WRITTEN IN ASSEMBLY LANGUAGE
EOF

1.3.6 MASS STORAGE FORTRAN 2.0B PAPER AND MAGNETIC TAPES

The installation material for FORTRAN 2.0B is on one magnetic tape and on ten paper tapes with each phase on a different paper tape. Source information is on one magnetic tape in COSY format. SELCOP and IOCAL are on one paper tape. There are three list magnetic tapes.


Installation Tapes
PAPER TAPE
PHASE A1
PHASE A2
PHASE A3
PHASE A4
PHASE A5
PHASE B1
PHASE C1
PHASE D1
PHASE E1
OBJECT-TIME LIBRARY
OR
MAGNETIC TAPE
PHASE A1
PHASE A2
PHASE A3
PHASE A4
PHASE A5
PHASE B1
PHASE C1
PHASE D1
PHASE E1
OBJECT-TIME LIBRARY
EOF
COSY Source
PAPER TAPE
SELCOP
IOCAL
MAGNETIC TAPE
PHASE A
PHASE B
PHASE C
PHASE D
PHASE E
PHASES A, B, C, D, E ASSEMBLY LANGUAGE PROGRAMS
OBJECT-LIBRARY PROGRAMS IN FORTRAN
OBJECT-LIBRARY PROGRAMS IN ASSEMBLY LANGUAGE
EOF

List Magnetic Tapes for 2.0B



COMPILER PROGRAMS WRITTEN IN FORTRAN

PHASE A
PHASE B

COMPILER PROGRAMS WRITTEN IN FORTRAN

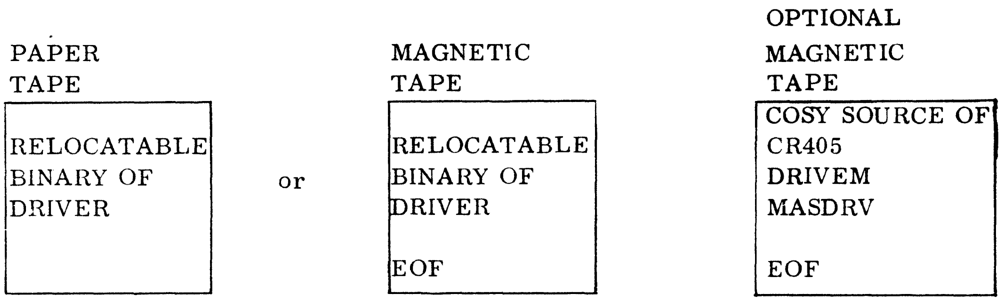
PHASE C
PHASE D
PHASE E

COMPILER PROGRAMS WRITTEN IN ASSEMBLY LANGUAGE

OBJECT-LIBRARY PROGRAMS WRITTEN IN FORTRAN

OBJECT-LIBRARY PROGRAMS WRITTEN IN ASSEMBLY LANGUAGE

1.3.7 1726-405 CARD READER DRIVER TAPE STRUCTURES



PAPER TAPE

RELOCATABLE BINARY OF DRIVER

or

MAGNETIC TAPE

RELOCATABLE BINARY OF DRIVER

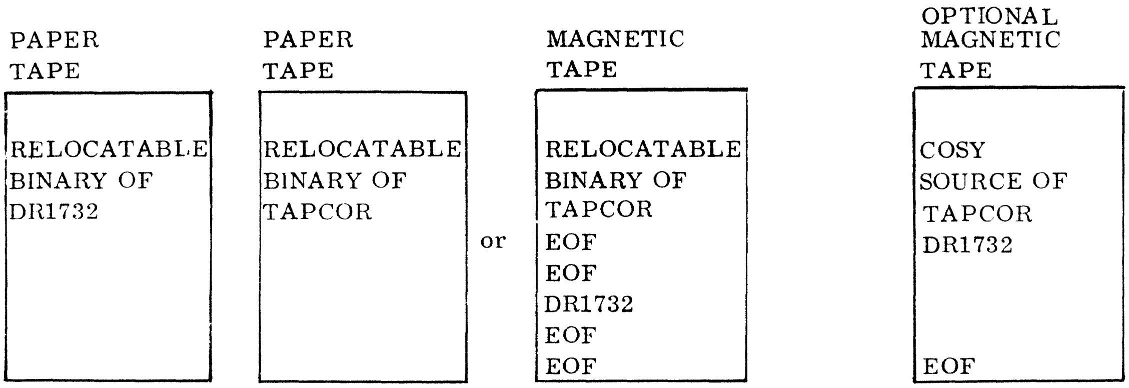
EOF

OPTIONAL MAGNETIC TAPE

COSY SOURCE OF
CR405
DRIVEM
MASDRV

EOF

1.3.8 1732-608/609 MAGENTIC TAPE DRIVER TAPE STRUCTURES



PAPER TAPE

RELOCATABLE BINARY OF DR1732

PAPER TAPE

RELOCATABLE BINARY OF TAPCOR

or

MAGNETIC TAPE

RELOCATABLE BINARY OF TAPCOR
EOF
EOF
DR1732
EOF
EOF

OPTIONAL MAGNETIC TAPE

COSY SOURCE OF
TAPCOR
DR1732

EOF

1.3.9 1740-501 LINE PRINTER DRIVER TAPE STRUCTURES

PAPER TAPES		MAGNETIC TAPES	OPTIONAL MAGNETIC TAPES
RELOCATABLE BINARY OF DRIVER	or	RELOCATABLE BINARY OF DRIVER EOF EOF	COSY SOURCE PRT40 DRIVEM MASDRV EOF EOF

1.3.10 SYSTEM CHECKOUT

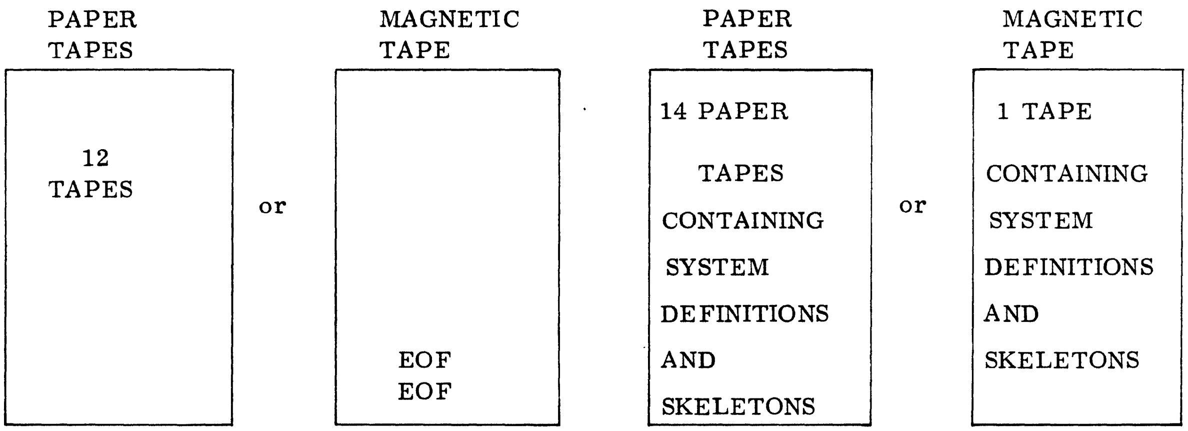
Installation Tapes

Optional Tapes

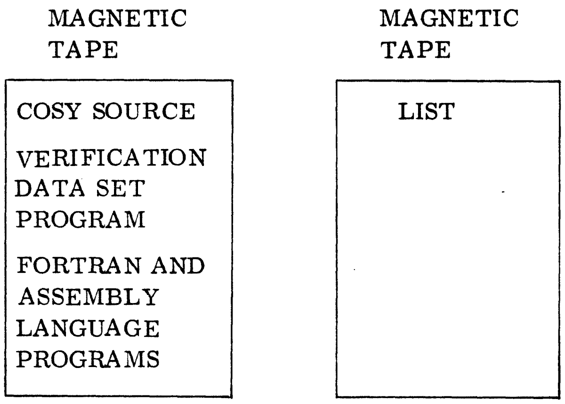
PAPER TAPES		MAGNETIC TAPE	MAGNETIC TAPE	MAGNETIC TAPE
COBOP SYSCOP CO1ST CO2ND CO3RD COLAST	or	COBOP EOF EOF SYSCOP EOF EOF CO1ST EOF EOF CO2ND EOF EOF CO3RD EOF EOF COLAST EOF EOF	COSY source of COBOP SYSCOP CO1ST CO2ND CO3RD COLAST EOF	LIST

1.3.11 SYSTEM CONFIGURATOR

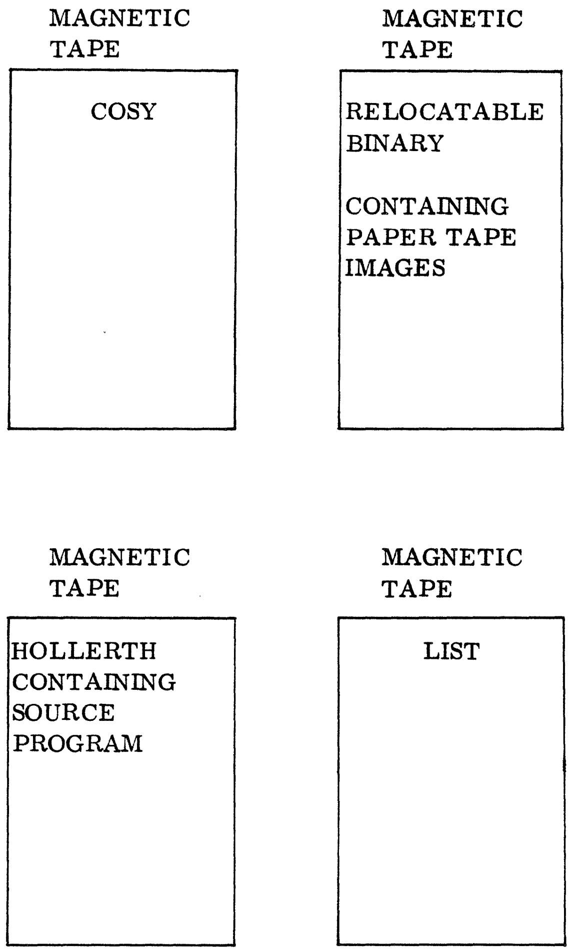
Installation Tapes


PAPER TAPES
12 TAPES
or
MAGNETIC TAPE
EOF
EOF
PAPER TAPES
14 PAPER TAPES CONTAINING SYSTEM DEFINITIONS AND SKELETONS
or
MAGNETIC TAPE
1 TAPE CONTAINING SYSTEM DEFINITIONS AND SKELETONS

Optional Tapes


MAGNETIC TAPE
COSY SOURCE
VERIFICATION DATA SET PROGRAM
FORTRAN AND ASSEMBLY LANGUAGE PROGRAMS
MAGNETIC TAPE
LIST

1.3.12 1777 PAPER TAPE STATION



1.4 NEW FEATURES

1.4.1 MSOS 2.1

The following changes have been made to the *M statement in LIBEDT:

*M, ordinal, sector, residence, flag

If flag is blank, an attempt will be made to patch remaining externals with a program library load

If flag is not blank, the attempt to patch remaining externals will not be made

1.4.2 MACRO ASSEMBLER 2.0

- Macro Assembler 2.0 processes macros at least 25% faster than Macro Assembler 1.2
- An OPT card may be used instead of typing in options on the teletypewriter
- The Ilu option, which is now part of the OPT card, allows the assignment of an input unit other than the standard system input unit
- The name of the program and the page number are printed on the top of each page of the listing
- Comment cards are permitted between the NAM and the MAC cards
- Diagnostic NN is produced if an assembly does not begin with a NAM image. If relocatable output is selected, a correct NAM block, including a blank name, is produced
- If a class 3 opcode (INP, OUT, ENA, etc.) has an address expression which is out of the range -127 to +127, the diagnostic EX is issued before the truncated value is placed in the lower 8 bits of the instruction
- Each of the passes has been divided into subprograms approximately 400_{16} in length. Each subprogram has a COSY sequence number in column 73-80 which will be used in answering PSR's on the Macro Assembler
- The diagnostic MO is produced if the assembler generated load-and-go file overflowed the load-and-go area on the disk
- A LB error is generated when illegal characters are in the label field

1.4.3 MASS STORAGE FORTRAN 2.0A

- Capability to do mixed mode arithmetic
- Compilation is approximately four times faster for version 1.1B
- When the output device is a printer, the source code now appears on the same line as the internal statement number

1.4.4 MASS STORAGE FORTRAN 2.0B

- Capability for mixed mode arithmetic
- Compilation approximately six times faster than for version 1.1B
- When the output device is a printer, the source code now appears on the same line as the internal statement number
- OPT card will select options from the standard input device or teletype

1.4.5 1728-430 READER/PUNCH DRIVER

This release is modified to include the EBCDIC option which is explained in Section 1.4.6.

1.4.6 1729-2 CARD READER DRIVER 2.1

This driver handles both the standard Hollerith to ASCII conversion (June 1966, communication of the ACM) and EBCDIC Hollerith to ASCII conversion (November 1968, communications of the ACM). The user may select the desired standard at driver assembly time. This option is controlled by an equate instruction which functions as follows:

EQU EBCDIC(0)	the table for the standard Hollerith to ASCII will be assembled
EQU EBCDIC(1)	the table for EBCDIC Hollerith to ASCII will be assembled

By changing this card, the user selects the desired table.

1.5 CORRECTIONS AND MODIFICATIONS

1.5.1 MSOS 2.1

All MSOS 2.0 PSR's received prior to December 1, 1968 have been included in MSOS 2.1. The numbers are:

197
212-217
219
222
228
229
240
241
243
244
249-252
254-273
275-277
280
282-284
287
289-294
298
301
304
305
307-309
311-313
317-323
325-332
336
338-344
351-355
359-361
363
365
367-369
374-384
386
397
417
424
426
427
440-443
446
450-452
457
463
466
473
476
477
482
486
494
500
504
505
510
511
519
520

1.5.2 MACRO ASSEMBLER 2.0

Self calls within a macro no longer result in an endless loop. Recursive calls are ignored after level of depth 10.

A source image which generates more than one line of code (ALF, EQU, BSS, etc.) does not cause the listing to go beyond the bottom on a page.

The 265th call to a user macro is now processed correctly.

An EX error on an external reference no longer results in the loss of that line in the listing.

Punching on 601 magnetic tape no longer causes a J06 error when backspacing over *T.

The VFD and DEC processors have been corrected so that they will process the statements sequentially.

An ALF statement which declares more characters than can be contained on the remainder of a 72 character image will not be processed.

The following are new diagnostics:

NN means missing name card.

LB means illegal characters in the label field.

MO means overflow of the load-and-go area.

1.5.3 MASS STORAGE FORTRAN 2.0A

The characters in columns 71-72 and 71-80 are now printed when the list option is selected on the teletype and printer, respectively.

Adding .8 and .2 gives the correct result.

Object-library entry points can now be declared as relative.

Hexadecimal output of +0 and -0 is now correctly formatted.

The following statement is now flagged as a non-fatal error and is processed as the product (I)*(K):

J = (I)(K)

Formatted write records are extended to 68 words (136 characters) so that the full capacity of the line printer is utilized.

BLOCK DATA programs compile correctly.

Rounding of a fractional part carrying into an integral part works correctly for the format FW.d when d=0.

The compiler generates proper code for a statement of the form:

Y = -X(I, J)

A statement of the following form now works correctly:

```
FM = PI/FLOAT (M-1)
```

Compiler options may be selected by OPT in columns 1-3 and the options given on the next card beginning in column 1. These precede the PROGRAM statement.

1.5.4 MASS STORAGE FORTRAN 2.0B

Corrections and modifications for Mass Storage FORTRAN 2.0B are the same as those listed in 1.5.3 for Mass Storage FORTRAN 2.0A.

1.5.5 1726-405 CARD READER DRIVER

An error in the DRIVEM routine prohibits the 405 mass memory resident driver from running correctly. To correct this problem incorporate the following corrective coding when DRIVEM is decompressed:

```
Label       Op          Address

DRIVEM      DCK/        I=6,H=7
            DEL/        21
            FRONT       MI1726,MC1726,ME1726
            END/
```

When using the unbuffered hardware conversion 1726-405 card reader driver, the separator card (6789 punch) is not recognized as an end of file card.

1.5.6 1740-501 LINE PRINTER DRIVER

Lines exceeding 136 characters cause the printer to fail and the job to hang. Therefore, insert the following corrective coding if print lines will exceed 136 characters:

```
PRT40       DCK/        I=6,H=7
            DEL/        188
            OUT         FULL-*
            END/
```

1.6 DEFICIENCIES AND LIMITATIONS

1.6.1 MSOS 2.1

Known Deficiencies

Downing a failed logical unit will periodically hang the system. See special PSR Summary 30 PSR 531 for corrective code.

Loading a program requiring more core than is available produces a J01 diagnostic in addition to the E5 loader diagnostic and causes further load aborts. See special PSR Summary 30 PSR 532 for corrective code.

A program load operation which generates an E3 diagnostic will also generate an E13 diagnostic.

The program deck names DMPCOR and MASDMP for the recovery module are identical to two deck names in the breakpoint module.

File marks on BCD tapes (even parity recording) are treated as parity errors. See special PSR Summary 30 PSR 533 for corrective code.

The first operation to a restored device is sometimes performed incorrectly. See special PSR Summary 30 PSR 534 for corrective code.

SBH command in ODEBUG does not work. See special PSR Summary 30, PSR 535 for corrective code.

A system hang results when downing a mag tape unit when another request is queued to another tape unit on the same controller. See special PSR Summary 30, PSR 536 for corrective code.

The 1729 card reader driver for the initializer is not usable as a system initializer driver and should be deleted from the initializer tape.

TAPDRB declares "JKIL" external and then never references it. See special PSR Summary 30, PSR 537 for the proper deletion.

Too long loader blocks input to the system initializer hang it.

*M statement of LIBEDT does not link to the CREP table. See special PSR Summary 30 PSR 538 for corrective code.

Known Limitations

The SCN command in ODEBUG does not reject illegal hexadecimal values but converts them to zero and continues.

Statement editing for errors within the system initializer is limited. Incorrect commands can cause initialization malfunctions which require restarting the process to alleviate the problem.

Incorrect formatting of output to the TTY will result if the output message buffering package is used with Standard Recovery.

An *P statement in LIBEDT punches a single all-ones frame on paper tape even when no valid input is received.

Programs TABLES, MIPROC, MEPROC, MMONI and MRW are provided to the user as a carry over from MSOS 2.0 with no additional development or testing since MSOS 2.0.

1.6.2 MACRO ASSEMBLER 1.2

Known Deficiencies

A user defined macro which will be used as input to LIBMAC must not contain any images with an * in column 1. A macro which is defined directly within a subprogram may have these images with no restriction.

Known Limitations

The Macro Assembler punches leader following the paper tape binary output from each program assembled but not preceding any load and go. Although this provides a separator between programs, it does not assure that leader will precede each program, especially the first program.

The assembler does not check for error conditions following completion of a request and thus may process invalid or improper data if the user returns control to the assembler following an I/O error. Unless incorrect data generates assembly diagnostics, disk errors are denoted by MASS MEMORY ERRORS only.

1.6.3 MASS STORAGE FORTRAN 2.0A

Known Deficiencies

The floating point package does not round properly on FW.d format

Runaway diagnostics result if the EQUIVALENCE table overflows. See PSR 528 in Summary 30.

Execution diagnostic 13 is repeated continually. See PSR 529 in Summary 30.

Execution diagnostic 5 is not given, but after writing an ENDFILE succeeding READ and WRITE requests to that unit are ignored. See PSR 527 in Summary 30.

Known Limitations

If superfluous information is included on an END line, the program is terminated but no diagnostic is given.

No check is made on the parameter type of the arguments of the intrinsic functions, the external functions, or the statement functions.

1.6.4 MASS STORAGE FORTRAN 2.0B

Known Deficiencies

The floating point package does not round properly on FW.d format.

Runaway diagnostics result if the EQUIVALENCE table overflows.
See PSR 528 in Summary 30.

Execution diagnostic 13 is repeated continually. See PSR 529 in Summary 30.

Execution diagnostic 5 is not given, but after writing an ENDFILE succeeding READ and WRITE requests to that unit are ignored. See PSR 527 in Summary 30.

Known Limitations

If superfluous information is included on an END line, the program is terminated but no diagnostic is given.

No check is made on the parameter type of the arguments of the intrinsic functions, the external functions, or the statement functions.

1.6.5 COSY 1.0

Known Limitations

With standard input assigned to the TTY, COSY still expects an input unit record of 80 characters.

1.6.6 1713 TELETYPEWRITER READER/PUNCH DRIVER

Known Deficiencies

When there is an alarm condition on the reader or punch and a request to the keyboard is queued in MASDRV, the system will hang. See special PSR summary 30, PSR 540 for corrective code.

1.6.7 1745-2 1.0 DISPLAY DRIVER

None.

1.6.8 SYSTEM CONFIGURATOR

Known Limitations

1. Logical unit numbers input on control statements must be in decimal and must be valid unit numbers for the system. SYSCON attempts to use any decimal number in the range of 1-127; thus, invalid unit numbers will cause SYSCON to terminate with a system JO2 error.

2. The restriction described in the first item above is also true for INPUT FROM LOGICAL UNIT parameter phrases associated with the INSERT components.

3. A comma must follow the component name even though no parameters are specified.

1.7 REQUIREMENTS

1.7.1 MSOS 2.1 REQUIREMENTS

Hardware Configuration

The minimum machine configuration for the 1700 Mass Storage Operating System is:

CONTROL DATA® 1704 Computer (with 4096 words of memory)

CONTROL DATA® 1705 Interrupt/Data Channel

CONTROL DATA® 1708 Storage Increment (2 with 4096 words of memory)

CONTROL DATA® 1711 Teletypewriter

CONTROL DATA® 1721 Paper Tape Reader or

  CONTROL DATA® 1722 Paper Tape Reader

CONTROL DATA® 1723 Paper Tape Punch or

  CONTROL DATA® 1724 Paper Tape Punch

CONTROL DATA® 1738 Disk Controller

CONTROL DATA® 853 Disk Drive or

  CONTROL DATA® 854 Disk Drive

Program operation can be enhanced by addition of other peripherals. Also, as peripherals are added and the system is expanded, the size of core storage must be expanded to accommodate the new drivers. Optional peripherals are listed below:

CONTROL DATA® 1706 Buffered Data Channel

CONTROL DATA® 1713 Teletypewriter

CONTROL DATA® 1729 Card Reader

CONTROL DATA® 1728-430 Card Reader/Punch

CONTROL DATA® 1729-2 Card Reader

CONTROL DATA® 1732 Magnetic Tape Controller

CONTROL DATA® 608 Magnetic Tape Transport

CONTROL DATA® 609 Magnetic Tape Transport

CONTROL DATA® 1742 Line Printer with Control

CONTROL DATA® 1745-2 Display Device

CONTROL DATA® 1751 Drum

CONTROL DATA® 1726-405 Card Reader Controller

CONTROL DATA® 1740 Printer Controller

CONTROL DATA® 501 Line Printer

Memory Requirements

All lengths are in decimal.

Normal Monitor:

Basic core resident	3529
Allocatable core	3000

Available Drivers:

Card Equipment	
1726-405 card reader	366
1728-430 card reader/punch	861
1729 card reader	357
1729-2 card reader	454
Disk or drum	
1738-853 disk	425
1751 drum	272
Display	
1745-2 buffered display	676
1745-2 unbuffered display	584
Line Printer	
1740-501 line printer	517
1742 line printer	478
Magnetic tape	
1731/1732 buffered magnetic tape	1086
1731-601 unbuffered magnetic tape	830
1732-608 unbuffered magnetic tape	830
1732-608/609 magnetic tape	348

Paper Tape	
1721/1722 paper tape reader	216
1723/1724 paper tape punch	207
Teletypewriter	
1711/1712/1713 teletypewriter	319
1713 reader/punch teletypewriter	594 + buffer size

1.7.2 MACRO ASSEMBLER 2.0

The largest core load of the assembler, PASS3, requires 3187_{10} plus 260_{10} words of common storage. The remainder of unprotected core is used to build the symbol table. If the length of the symbol table exceeds the length of remaining core, the symbol table is dumped out to mass storage.

1.7.3 MASS STORAGE FORTRAN 2.0A

Hardware Requirements

The minimum hardware configuration is 8K more core memory than that for MSOS 2.1.

Memory Requirements

Mass Storage FORTRAN 2.0A runs in 20K

instructions	5788
labeled common (data)	1649
blank common	1236
largest overlay	8673

Examples:

Paper Tape System:

Monitor (Disk plus TTY)	7273
1721/1722	216
1723/1724	207
FORTRAN 2.0A	8673
	16,369 words of memory

Card System with Printer:

Monitor (Disk plus TTY)	7273
1728-430	861
1742	478
FORTRAN 2.0A	8673
	17285 words of memory

Card/Tape System with Printer:

Monitor (Disk plus TTY)	7273
1728-430	861
1742	478
1731-601 unbuffered	830
FORTRAN 2.0A	8673
	18115 words of memory

Mixed System:

Monitor (Disk plus TTY)	7273
1721/1722	216
1723/1724	207
1728-430	861
1742	478
1731-601 buffered	830
FORTRAN 2.0A	8673
	18538 words of memory

1.7.4 MASS STORAGE FORTRAN 2.0B

Hardware Requirements

The minimum hardware configuration is the same as that for MSOS 2.1.

Memory Requirements

Version 2.0B is designed to run in 24K.

instructions	9800
labeled common (data)	1649
blank common	1236
largest overlay	12,685

1.7.5 COSY

COSY requires 2829_{10} words of unprotected core to execute.

1.7.6 SYSTEM CHECKOUT

Hardware Requirements

The minimum hardware configuration is the same as that for MSOS 2.1.

Memory Requirements

System Checkout uses no more than 500 words of core memory.

1.7.7 SYSTEM CONFIGURATOR

Hardware Requirements

The minimum hardware configuration is the same as that for MSOS 2.1. For optimum installation and execution add the following hardware:

Either 2 1731/601 magnetic tape units or
2 1732/608 magnetic tape units

Either 1 1726-405 card reader or
1 1728-430 card reader or
1 1729-2 card reader and 1 1742 line printer

Software Requirements

A minimum of 3000 words of unprotected core is necessary to execute SYSCON.

1.8 PUBLICATIONS

1700 OPERATING SYSTEM OPERATING GUIDE	60191400
1700 MSOS REFERENCE MANUAL	60223100C
1700 COMPUTER SYSTEM MACRO ASSEMBLER REFERENCE MANUAL	60176300A
1700 COMPUTER SYSTEM MASS STORAGE/FORTRAN REFERENCE MANUAL	60192200A
1700 COSY/MSOS REFERENCE MANUAL	60237100
1700 CONTROL DATA 1700 COMPUTER SYSTEM CODES	60163500
1700 SYSTEM CHECKOUT REFERENCE MANUAL	60281800
1700 SYSTEM CONFIGURATOR REFERENCE MANUAL	60282300A

PART II

INSTALLATION PROCEDURES

PREDEFINED AND CAPSULIZED PROCEDURES 1

1.1 PREDEFINED PROCEDURES

1.1.1 ENTERING DATA INTO CORE MEMORY

1. MASTER CLEAR
2. Set all switches to the neutral positions
3. Set SELECTIVE STOP switch
4. Set P register
5. Set push button register to the core location into which the first word is to be stored
6. Set ENTER/SWEEP switch to ENTER
7. Set X register
8. Enter first (or next) word of code into push button register
9. Momentarily move STEP/RUN switch to STEP
10. Clear the X register by pressing CLEAR
11. Repeat steps eight through ten for all words to be entered
12. Release SELECTIVE STOP switch when finished

1.1.2 EXAMINING DATA IN CORE MEMORY

1. MASTER CLEAR
2. Set all switches to the neutral positions
3. Set SELECTIVE STOP switch
4. Set the P register
5. Set the push button register to the first core location to be examined
6. Set the X register
7. Set the ENTER/SWEEP switch to SWEEP
8. Momentarily move the STEP/RUN switch to STEP
9. The data in the core location entered into the P register above will be displayed on the push button register
10. Repeat step eight to display the next sequential word of core memory
11. Release SELECTIVE/STOP switch when finished

1.1.3 EXECUTING INSTRUCTION SEQUENCE

1. MASTER CLEAR
2. Set all switches to neutral position
3. Set the P register
4. Enter the core location for the first instruction of the sequence into the push button register
5. Set the A, Q, and X registers to their specified contents
6. Set the SELECTIVE STOP and/or the SELECTIVE SKIP switches if necessary
7. Move the STEP/RUN switch to RUN momentarily

1.2 CAPSULIZED PROCEDURES

Capsulized installation procedures summarize the steps necessary to install the system initializer and the 2.1 operating system. They are designed for the user who has complete familiarity with the detailed installation procedures. System Initializer messages are listed in section III. 2. 2. 1.

1. Mount a disk pack on the disk drive
2. Manually load the bootstrap instruction sequence by entering the followi~g paper tape loading sequence into core beginning at location 200 and using the X register:

```
6818   0DFE   02FE   02FE   0102   0000
0A20   02FE   680C   6C08   D803   0000
E000   0111   0A00   D807   18F6
00A1   18FD   02FE   C805   18FF
03FE   0FC8   0FC8
```

3. Check number of entries by selecting the P register. The display should be 218
4. Examine data in core memory using instructions in 1.1.2
5. Read the checksum loader by:
    a. Mounting system initializer paper tape in paper tape reader (the checksum loader is on the front of this tape)
    b. Releasing ENTER/SWEEP to neutral position
    c. Pressing MASTER CLEAR on paper tape reader
    d. Setting A and Q registers to 0000
    e. Setting P register to 0200
    f. Switching STEP/RUN to RUN
6. When the checksum loader is read the tape stops
7. Execute the checksum loader by:

a. Setting the RUN/STEP switch to STEP and MASTER CLEAR

b. Setting P and Q registers to 0000

c. Setting A register to xxxx which is the address to which the system initializer is to go in core. xxxx is the length of MAXCOR minus the initializer length ($157D_{16}$)

d. Setting SELECTIVE STOP switch (other switches should be neutral)

e. After setting STEP/RUN switch to RUN, the tape loads and stops

f. Selecting the Q register; display should read 0000. If the Q register does not show 0000, a checksum error occurred

g. Releasing SELECTIVE STOP

h. MASTER CLEAR

8. Execute system initializer by:

a. Setting P register to xxxx which is the system initializer address

b. Switching STEP/RUN to RUN

9. Message: SI
This indicates that the system initializer can now load the operating system

10. Mount the first installation tape in the paper tape reader or mount the corresponding magnetic tape on the magnetic tape unit

11. Type: *S, MAXCOR, xxxx
xxxx is the highest core location used by the system

xxxx	System Size
2FFF	12K
3FFF	16K
4FFF	20K
5FFF	24K
6FFF	28K
7FFF	32K

To reserve areas in upper core for permanent bootstrap loaders and/or core dump programs, set MAXCOR to less than the system core size.

Press: CARRIAGE RETURN

12. Type: *S, SECTOR, xxxx

xxxx is the maximum number of sectors to be used by disk pack system

xxxx	Unit
AA9	1751E
1552	1751J
3E7F	1738-853
7CFF	1738-854
2FFF	For mass memory buffering when using software buffering package

At times it is desirable to limit system scratch by setting SECTOR to a value less than the two maximums mentioned above for the 853 and the 854 disk drives. Reducing the system scratch area reduces the length of seeks during assemblies and compilations and provides a file area accessible to the user only.

Press: CARRIAGE RETURN

13. If necessary delete drivers. See 2.2, step 5 for details.

    Type: *S, driver entry point, 7FFF

14. Reassign input if it is not to be paper tape reader. See 2.2, step 6.
15. Type: *V
16. Press: CARRIAGE RETURN

    If unpatched externals result at the end of either an *M load, or an *L load, or at the end of system initialization following an *T command, an ERROR C or ERROR D appears on the system initialization comment device.

    To continue initialization, repeat the last control statement typed (either *M or *L load commands or the *T).

    ```
                  *M
    Type: either  *L
                  *T
    ```

    Press: CARRIAGE RETURN

17. If unpatched externals are not present, or if the action in step 16 is taken, the sector address of core image appears and the following message is typed.

    ```
    Message: TIMER RJ        (if no timer is present on system)
             PP              (indicates that the operator should set PROGRAM PROTECT
                             switch)
    ```

18. Switch the PROGRAM PROTECT switch up.

    Type: *

    Press: CARRIAGE RETURN

    Press: MANUAL INTERRUPT

19. Type: *LIBEDT

    Press: CARRIAGE RETURN

20. Type: *V, lu

    lu is the logical unit number of the input device.

    Press: CARRIAGE RETURN

21. Typeout appears followed by

    Message: IN

    Press: CARRIAGE RETURN

22. Type: *Z

    Message: J

    The operating system is installed.

DETAILED OPERATING SYSTEM INSTALLATION PROCEDURES 2

2.1 LOADING OF SYSTEM INITIALIZER

2.1.1 MANUAL LOADING OF BOOTSTRAP

System Initializer messages are listed in section III. 2. 2. 1.

Entering the Paper Tape Loading Sequence

Enter the loading sequence into core memory beginning at core memory location 200. This code can be loaded at any location or run anywhere above the last location into which the checksum loader will load, but the location 200 is preferable. This sequence of code will read one formatted record (the checksum loader) into the location specified by the A register (which will be 0000).

1. MASTER CLEAR
2. Set all switches to neutral positions
3. Set SELECTIVE STOP switch
4. Set P register
5. Set push button register to 200
6. Set ENTER/SWEEP switch to ENTER
7. Set X register
8. Enter the code in this manner:
    a. Enter first (or next word) of code into push button register
    b. Momentarily set STEP/RUN switch to STEP
    c. Clear the push button register
    d. Proceed with each word using steps a through c until all code is entered

    Code:

```
6818     0FC8     D807
0A20     02FE     C805
E000     680C     0102
00A1     0A00     D803
03FE     02FE     18F6
0DFE     0FC8     18FF
02FE     02FE     0000
0111     6C08     0000
18FD
```

9. Set the P register
10. The display should show 218_{16} which means that 25_{10} commands have been entered
11. Release SELECTIVE STOP

Optional Checking of Loading Sequence

1. MASTER CLEAR
2. Set all switches to their neutral position
3. Set SELECTIVE STOP switch
4. Set P register
5. Enter into the push button register the first core location to be examined
6. Set X register
7. Set ENTER/SWEEP switch to SWEEP position
8. Momentarily set the STEP/RUN switch to the STEP position

   The data in the core location specified in step 5 appears on the push button register
9. To display the next sequential word of core memory in the push button register, briefly set the STEP/RUN switch to the STEP position
10. Release SELECTIVE STOP

2.1.2 READING CHECKSUM LOADER

1. Mount the MSOS 2.1 system initializer paper tape on the paper tape reader. The checksum loader is on the front part of this tape
2. Set all switches to neutral
3. MASTER CLEAR
4. Set the P register button
5. Set push button register to 0200
6. Set the STEP/RUN switch to RUN

   The first few feet (checksum loader) are read from the tape into core memory at location 0000, the tape then stops.

2.1.3 EXECUTING CHECKSUM LOADER

1. Position the system initialization tape in the paper tape reader
2. MASTER CLEAR
3. Set the A register
4. Set the push button register to xxxx

   xxxx is the length of MAXCOR minus the initializer length ($157D_{16}$)
5. Set the SELECTIVE STOP switch

6. Set the STEP/RUN switch to RUN to load the tape
7. When the tape stops, set the Q register
8. The push button register should be 0000
9. If the push button register does not read 0000, a checksum error occurred
   a. Re-insert the initializer portion of the tape into the reader
   b. Return to step 2

2.1.4 EXECUTING SYSTEM INITIALIZER

1. MASTER CLEAR
2. Release SELECTIVE STOP switch.
3. Set P register
4. Set the push button register to xxxx which is the address of the system initializer
5. Momentarily set the STEP/RUN switch to RUN
6. SI appears on the teletypewriter to indicate that the system initializer can now load the operating system

2.2 OPERATING SYSTEM INSTALLATION

1. Mount the first MSOS 2.1 installation paper tape in the paper tape reader or mount the corresponding magnetic tape on the magnetic tape unit. Set to equipment number seven, unit 0
2. Type: *S, MAXCOR, xxxx

   xxxx is the highest core location used by the system

xxxx	System Size
2FFF	12K
3FFF	16K
4FFF	20K
5FFF	24K
6FFF	28K
7FFF	32K

   To reserve areas in upper core for permanent bootstrap loaders and/or core dump programs, set MAXCOR to less than the system core size

   Press: CARRIAGE RETURN

   Message: Q

3. Type: *S, SECTOR, xxxx

   xxxx indicates the maximum number, in hexadecimal, of disk pack sectors to be used by the operating system

xxxx	Unit
AA9	1751E
1552	1751J
3E7F	1738-853
7CFF	1738-854
2FFF	for mass memory buffering when using software buffering package

   Press: CARRIAGE RETURN

   Message: Q

   At times it is desirable to limit system scratch by setting SECTOR to a value less than the two maximums mentioned above for the 853 and the 854 disk drives. Reducing the system scratch area reduces the length of seeks during assemblies and compilations and provides a file area accessible to the user only.

4. To add drivers, see 3.6 DRIVER ADDITION

5. To reduce the size of the core resident system, delete unnecessary drivers at this point in installation

   Type: *S, entry point, 7FFF

   Press: CARRIAGE RETURN

   Message: Q

   Entry points for the various standard drivers are listed below. Even though only one entry point is listed for each driver, any entry point may be used.

Driver	Entry Point
1711/1712/1713 teletypewriter	TYPI
1721/1722 paper tape reader	PTREAD
1723/1724 paper tape punch	PUNCDR
1726-405 card reader	CR405
1728-430 card reader	IN1728
1729 card reader	CARDI
1729-2 card reader	IN1729
1731/1732 unbuffered magnetic tape control	TAPEDR
1731/1732-1706-601/608 buffered magnetic tape control	TAPDRB
1731/1732 recovery	RECOVT
1731-1706 recovery	RECVTB
1731/1732 tape motion control	T14
1731/1732-601/608 format ASCII read/write	FRWA

Driver	Entry Point
1731/1732-1706-601/608 buffered format ASCII read/write	FRWAB
1731/1732-601/608 format binary read/write	FRWB
1731/1732-601/608 buffered format binary read/write	FRWBB
1731/1732-601/608 non-format read/write	RWBA
1731/1732-1706-601 /608 buffered non-format read/write	RWBAB
1731/1732-609 ASCII binary read/write	RW609
1731/1732-1706-609 buffered format ASCII binary read/write	RW609B
1732-608/609 buffered/unbuffered formatted/unformatted read/write	DR1732
1738-853/854 disk	DISK
1738-853/854 disk word	DISKWD
1740-501 line printer	PRT40
1742 line printer	PRINTI
1751 drum	DRMDRZ
1745-2 display	DDINIT

Examples:

To delete the printer driver:

Type: *S, PRINTI, 7FFF

Press: CARRIAGE RETURN

Message: Q

To delete the unbuffered magnetic tape driver, type all non-buffered driver names:

Type: *S, TAPEDR, 7FFF

Press: CARRIAGE RETURN

Message: Q

Type: *S, FRWA, 7FFF

Press: CARRIAGE RETURN

Message: Q

Type: *S, FRWB, 7FFF

Press: CARRIAGE RETURN

Message: Q

Type: *S, RECOVT, 7FFF

Press: CARRIAGE RETURN

Message: Q

Type: *S, T14, 7FFF

Press: CARRIAGE RETURN

Message: Q

During initialization the printout includes an error 17 message for each of the drivers deleted with an *S.

6. Initializing from other media:

The system initializer is initially set to accept input from a paper tape reader, output to disk, and list on the teletypewriter. A 1711/1712/1713 teletypewriter is assumed to be the comment I/O device.

If the initial input was from the paper tape reader and the operating system is to be built from another device, reassign units at this time. See Part III, Section 2.1 for additional initializer control statements and Part III, Section 2.2 for initializer diagnostics.

Reassignment:

To Reassign the Input Device:

Type: *I, lun

lun = 1 1721/1722 paper tape reader

3 1731/1732-601/608 magnetic tape unit (equipment = 7, unit = 0)

Press: CARRIAGE RETURN

Message: Q

To Reassign the Output Device:

Type: *0, lun

lun = 4 1738-853/854 Disk Pack

5 1751 Drum

Press: CARRIAGE RETURN

Message: Q

To Reassign the Comment and List Device:

Type: *C, lun

lun = 6 1711/1712/1713 Teletypewriter

7 1742 Line Printer

8 Dummy List Device

Press: CARRIAGE RETURN

Message: Q

All system initializer messages appear on the comment device with the maps

7. Type: *V

   Press: CARRIAGE RETURN

8. The tape is read.

   The program names on the tape are typed on the list device.

   If using paper tapes, the following message appears after each of the installation tapes are read:

   Message: L, lun FAILED.

   ACTION

   Mount the next installation paper tape on the paper tape reader.

   Press: READY MASTER CLR on the paper tape reader

   Type: RP

   Press: CARRIAGE RETURN

   During system initialization, the printout is described as follows:

   Format: name xxxx

   name The program name

   xxxx First word address (FWA) for core resident (*L) programs

   Beginning sector number of the groups of programs associated with the *YM ordinal for mass memory (*M) resident programs

   If unpatched externals result at the end of either an *M load, or an *L load, or at the end of system initialization following an *T command, an ERROR C or ERROR D appears on the system initialization comment device.

   To continue initialization, repeat the last control statement typed (either *M or *L load commands or the *T).

   Type: either *M
   *L
   *T

   Press: CARRIAGE RETURN

   The list output during initialization is as follows:

```
*S, ONE, 7FFF
*S, TWO, 7FFF
*S, THREE, 7FFF
*YM, LOADSD, 1, JOBENT, 2, JOBPRO, 3, JPLOAD, 4, JPST, 5
*YM, JPCHGE, 6, JBKILL, 7, JPT13, 8, MIPRO, 9, LIBEDT, 10
*YM, MOD1, 11, MOD2, 12, MOD3, 13, MOD4, 14, RESTOR, 15
```

```
*YM, ODEBUG, 16, RCOVER, 17, BRKPT, 18
*L        LOCORE
   LOCORE    0000
   SYSBUF    0109
   SCHEDU    05E6
   NDISP     0685
   NCMPRQ    06C1
   NFNR      06F2
   ADEV      075C
*M        LOADER
   LOAD      0001
   BRANCH    0001
   LIDRIV    0001
   LCDRIV    0001
   LMDRIV    0001
   LLDRIV    0001
   SCAN      0001
   CHPU      0001
   ADJOVE    0001
   CONVRT    0001
   TABSCH    0001
   TABSTR    0001
   LSTOUT    0001
   LINK1     0001
   LINK2     0001
   COREXT    0001
   DPRADD    0001
   LOADER    0001
   NAMPRO    0001
   RBDBZS    0001
   ENTEXT    0001
   XFRPRO    0001
   HEXPRO    0001
   EOLPRO    0001
   ADRPRO    0001
*L        DRCORE
   DRCORE    089D
   ALCORE    09D2
   ALVOL     0A7B
   OFVOL     0A98
   TRVEC     0AA4
   PARAME    0AC1
   COMMON    0B1F
   NIPROC    0B36
   NEPROC    0BB2
   NMONI     0C16
   RW        0C58
   MAKQ      0CF4
   MINT      0D17
```

```
*M      JOBENT
   JOBENT    0021
   T11       0021
   T7        0021
   T3        0021
*M      JOBPRO
   JOBPRO    0025
   PROTEC    0025
   T5        0025
*M      JPLOAD
   JPLOAD    0033
*M      JPST
   JPST      0038
*M      JPCHGE
   JPCHGE    003A
   ASCHEX    003A
*M      JBKILL
   JBKILL    003E
*M      JPT13
   JPT13     0040
   T13       0040
*M      MIPRO
   MIPRO     0046
*M      LIBEDT
   LIBEDT    0049
*M      UTILIB
   UTILIB    0054
*M      PLINSN
   PLINSN    0061
*M      FILE
   FILE      006E
*M      GENLIB
   GENLIB    007C
*M      RESTORE DEVICE
   RESTOR    0082
*M      ODEBUG
   ODEBUG    0085
*M      RCOVER
   RCOVER    009A
   OUTSEL    009A
   DMPCOR    009A
   MASDMP    009A
*M      BRKPT
   BRKPTD    00A3
   SIFT      00A3
   BIASCI    00A3
   RETJMP    00A3
   JUMPTO    00A3
   ENTER     00A3
   ENTCOR    00A3
   PRTREG    00A3
   TERMIN    00A3
   RESUME    00A3
   DMPCOR    00A3
   MASDMP    00A3
   SETBRP    00A3
*L       DRIVERS
   DR1728    0DCB
   CD1729    1128
   PTREAD    12EE
   PUNCDR    13C6
   TELTYP    1495
   TAPEDR    15D5
   FRWA      170B
   FRWB      17C5
   RECOVT    1897
   TAPE      1909
   CARDRD    1913
   PRINTR    1A78
   DISKWD    1C56
   SPACE     1E00
*S, TIMINT, 7FFF
*S, SNAPE, 7FFF
*S, PARITY, 7FFF
*S, IPROC1, 7FFF
*S, T30, 7FFF
*S, T29, 7FFF
*S, T28, 7FFF
*S, T27, 7FFF
*S, T26, 7FFF
*S, T25, 7FFF
*S, T24, 7FFF
*S, T23, 7FFF
*S, T22, 7FFF
*S, T21, 7FFF
*S, T20, 7FFF
*S, T19, 7FFF
*S, T18, 7FFF
*S, T17, 7FFF
*S, T16, 7FFF
*S, T13, 7FFF
*S, T11, 7FFF
*S, T8, 7FFF
*S, T7, 7FFF
*S, T5, 7FFF
*S, T3, 7FFF
```

These are unpatched externals (entry points of programs not present in the normal system). To prevent an error printout, they are linked to 7FFF. If any of these modules are to be used, the *S statement associated with it should be deleted.

```
*S, JKIL, 7FFF
*S, RWBA, 7FFF
*S, RW609, 7FFF
*S, DEBUG, 7FFF
*S, DTIMER, 7FFF
*S, MAS300, 7FFF
*T
00EC
  IN
```

These are unpatched externals (entry points of programs not present in the normal system). To prevent an error printout, they are linked to 7FFF. If any of these modules are to be used, the *S statement associated with it should be deleted.

10. The sector onto which the core image for the new system was written is output on the assigned comment device. The following message appears if there is no timer in the hardware configuration or if there is a timer which rejected.

    Message: TIMER RJ
    PP

11. Set the PROGRAM PROTECT switch up.

12. Type: *

    Press: CARRIAGE RETURN

    Press: MANUAL INTERRUPT on the teletypewriter

    Message: MI

13. Type: *LIBEDT

    Press: CARRIAGE RETURN

    Message: LIB
    IN

14. Type: *V, lun

    lun is the logical unit number of the device which contains the input

    Press: CARRIAGE RETURN

    Message: (on the standard print device)

```
IN
*S, 1, 0, M
IN
*S, 2, 1, M
IN
*S, 3, 2, M
IN
*S, 4, 3, M
IN
*S, 5, 3, M
IN
*S, 6, 3, M
IN
*S, 7, 3, M
```

```
IN
*S,8,3,M
IN
*S,9,4,M
IN
*S,10,2,M
IN
*S,11,3,M
IN
*S,12,3,M
IN
*S,13,3,M
IN
*S,14,3,M
IN
*S,15,4,M
IN
*S,16,5,M
IN
*S,17,2,M
IN
*S,18,0,M
IN
*U
IN
```

The LIBEDT operation fixes the request priorities of the mass memory resident programs which insures proper allocation of core.

15. To sign off LIBEDT

    Type: *Z

    Press: CARRIAGE RETURN

    Message: J

    The job processor is now in core; normal operations may continue.

2.3 MACRO ASSEMBLER 2.0 INSTALLATION

2.3.1 REQUIREMENTS

1. MSOS 2.1 operating system must already be installed.
2. Disk is the scratch area for both the Macro Assembler and the load-and-go information.
3. The MSOS parameter SECTOR defines the maximum sector address which the Macro Assembler may use.
4. Memory requirements are defined in I.1.7.2.

2.3.2 INSTALLATION PROCEDURES

1. Type: *LIBEDT

   Press: CARRIAGE RETURN

   Message: LIB
   IN

2. If using magnetic tape:

   a. Mount the Macro Assembler release installation magnetic tape on LU 6

   b. When READY lights, type: *V, 06

   c. Press: CARRIAGE RETURN

   If using paper tape:

   a. Mount Macro Assembler release installation paper tape on LU 2

   b. Press: MASTER CLEAR on paper tape reader

   c. Type: *V, 02

   d. Press: CARRIAGE RETURN

   LIBEDT installs the Macro Assembler in the program library and generates the following listing on the list device:

```
IN
*K, I6, P8
IN
*L, ASSEM
IN
*P, F
    PASS1          nnnn†

    PA1 PR2        nnnn†
IN

*K, I8
IN
*N, PASS1,,, B
IN
*K, I6
IN
*P, F
    PASS2          nnnn†
    PA2 PR2        nnnn†
IN
*K, I8
IN
*N, PASS2,,, B
```

†nnnn load occurs at this address

```
IN
*K, I6
IN
*P, F
     PASS3              nnnn[†]
     PA3PR2             nnnn[†]
     PA3PR3             nnnn[†]
IN
*K, I8
IN
*N, PASS3,,, B
IN
*K, I6
IN
*P, F
     PASS4              nnnn[†]
IN
*K, I8
IN
*N, PASS4,,, B
IN
*K, I6
IN
*N, MACSKL,,, B
IN
*N, MACROS,,, B
IN
*U
```

3. Type: *Z

   Press: CARRIAGE RETURN

   Message: J

4. Macro Assembler 2.0 is installed and is ready to assemble source program

2.4 COSY 1.0 INSTALLATION

2.4.1 REQUIREMENTS

MSOS 2.1 must be installed. For memory requirements, see I.1.7.5.

2.4.2 INSTALLATION PROCEDURES

1. Type: *LIBEDT

[†] nnnn load occurs at this address

Press: CARRIAGE RETURN

Message: LIB
IN

2. Mount the relocatable binary tape

3. Assign lu to the device which contains the input

4. Type: *K, Ilu

   Press: CARRIAGE RETURN

5. Type: *L, COSY

   Press: CARRIAGE RETURN

   LIBEDT responds when loading is completed

   Message: IN

6. Type: *Z

   Press: CARRIAGE RETURN

   Message: Q

   COSY is now on the program library and is ready to execute

2.5 MASS STORAGE FORTRAN 2.0A AND 2.0B INSTALLATION PROCEDURES

2.5.1 REQUIREMENTS

1. MSOS 2.1 must already be installed

2. For memory requirements, see I.1.7.3 and 1.7.4

3. The logical unit numbers must be:

   lun 8 for mass storage device

   lun 6 for magnetic tape device

   lun 2 for paper tape reader with the standard install materials

2.5.2 INSTALLATION PROCEDURES

1. Type: *LIBEDT

   Press: CARRIAGE RETURN

   Message: LIB
   IN

2. If using magnetic tape:

   a. Mount the installation magnetic tape on the magnetic tape device

b. Set the Unit Select Wheel to 0 (LUN 6)

c. Press: LOAD

d. Press: READY

e. Type: *V, 06

f. Press: CARRIAGE RETURN

g. Message: IN

If using paper tape:

a. Mount paper tape 1 (Phase A1) on paper tape reader (LUN 2)

b. Press: READY MASTER CLR

c. Type: *V, 02

d. Press: CARRIAGE RETURN

e. Message: IN

f. Place next tape in paper tape reader

g. Type: *V, 02

Press: CARRIAGE RETURN

3. Output for 2.0A and 2.0B

The following output appears on the standard list device during the installation of FORTRAN 2.0A and 2.0B on the program library. When paper tape is used, *K, I2, P8 appears instead of *K, I6, P8.

2.0A

```
IN

*K, I6, P8
IN

*P
     FTN         2991
     GOA         3043
     CNVT        3087
     CONV        30C5
     DIAG        30F8
     EXP9        3188
     FLOAT       3235
     GETSYM      337F
     GPUT        33B8
     IOPRBA      33E1
     PACK        35D1
     Q8PRMS      35F6
     STORE        3607
     SYMBOL       3635
     LOCLA1       36DC
     DUMYA1       378A
     ENDDO        37F1
     GETC         38F2
     GETF         390B
     GNST         3BDB
     IGETCF       3D7E
     OPTION       3D97
     OUTENT       3DD7
     PHASEA       3E06
     PLABEL       42E6
     Q8QBDS       433C
     RDLABL       433C
     STCHAR       438A
     TYPE         43BC
     ENDLOC       45BC
IN

*K, I8
IN

*N, FORTA1,,, B
IN

*K, I6, P8
IN

*P
     FTN          2991
     GOA          3043
     CNVT         3087
     CONV         30C5
     DIAG         30F8
     EXP9         3188
     FLOAT        3235
     GETSYM       337F
     GPUT         33B8
     IOPRBA       33E1
     PACK         35D1
     Q8PRMS       35F6
     STORE        3607
     SYMBOL       3635
     LOCLA2       36DC
     DUMYA2       378D
     ARITH        379A
     COMNPR       3DE0
     DIMPR        3E76
     GETC         3FFD
```

```
        GETF          4016
        SUBSCR        42E6
        TYPEPR        45A2
        ENDLOC        45B9
IN

*K, I8
IN

*N, FORTA2,,, B
IN

*K, I6, P8
IN

*P
        FTN           2991
        GOA           3043
        CNVT          3087
        CONV          30C5
        DIAG          30F8
        EXP9          3188
        FLOAT         3235
        GETSYM        337F
        GPUT          33B8
        IOPRBA        33E1
        PACK          35D1
        Q8PRMS        35F6
        STORE         3607
        SYMBOL        3635
        LOCLA3        36DC
        DUMYA3        378D
        BYEQPR        379A
        CHECKF        398C
        CONSUB        3A2C
        DATAPR        3AB3
        FGETC         3C43
        FORK          3C62
        GETC          3DD9
        GETF          3DF2
        STCHAR        40C2
        TREE          40F4
        ENDLOC        45F1
IN

*K, I8
IN

*N, FORTA3,,, B
IN
```

```
*K, I6, P8
IN

*P
      FTN           2991
      GOA           3043
      CNVT          3087
      CONV          30C5
      DIAG          30F8
      EXP9          3188
      FLOAT         3235
      GETSYM        337F
      GPUT          33B8
      IORRBA        33E1
      PACK          35D1
      Q8PRMS        35F6
      STORE         3607
      SYMBOL        3635
      LOCLA4        36DC
      DUMYA4        378D
      ARAYSZ        3794
      ASGNPR        37FE
      BDOPR         3844
      CFIVOC        397E
      CKIVC         39DC
      CKNAME        39EC
      CPLOOP        39FC
      ENDDO         3AA1
      GETC          3BA2
      GETF          3BBB
      IOSPR         3E8B
      OUTENT        4519
      RDLABL        4348
      STCHAR        4596
      ENDLOC        45C8
IN

*K, I8
IN

*N, FORTA4, , , B
IN

*K, I6, P8
IN

*P
      FTN           2991
      GOA           3042
      CNVT          3087
      CONV          30C5
```

```
    DIAG        30F8
    EXP9        3188
    FLOAT       3235
    GETSYM      337F
    GPUT        33B8
    IOPRBA      33E1
    PACK        35D1
    Q8PRMS      35F6
    STORE       3607
    SYMBOL      3635
    LOCLA5      36DC
    DUMYA5      378A
    ARITH       3797
    GETC        3DE2
    GETF        3DFB
    SUBSCR      40CB
    ENDLOC      4387
IN

*K, I8
IN

*N, FORTA5,,, B
IN

*K, I6, P8
IN

*P
    FTN         2991
    GOA         3043
    CNVT        3087
    CONV        30C5
    DIAG        30F8
    EXP9        3188
    FLOAT       3235
    GETSYM      337F
    GPUT        33B8
    IOPRBA      33E1
    PACK        35D1
    Q8PRMS      35F6
    STORE       3607
    SYMBOL      3635
    LOCLA6      36DC
    DUMYA6      378D
    CFIVOC      3794
    CKIVC       37F2
    ERBPR       3802
    GETC        3855
    GETF        386E
```

```
     MODMXR       3B3E
     RDLABL       3F95
     SUBPPR       3FE3
     TREE         40A3
     ENDLOC       45A0

IN

*K, I8
IN

*N, FORTA6,,, B
IN

*K, I6, P8
IN

*P
     FTN          2991
     GOA          3043
     CNVT         3087
     CONV         30C5
     DIAG         30F8
     EXP9         3188
     FLOAT        3235
     GETSYM       337F
     GPUT         33B8
     IOPRBA       33E1
     PACK         35D1
     Q8PRMS       35F6
     STORE        3607
     SYMBOL       3635
     LOCLA7       36DC
     DUMYA7       378D
     ASEMPR       378D
     EXRLPR       3937
     GETC         3995
     GETF         39AE
     IGETCF       3C7E
     PEQVS        3C97
     PRNTNM       4076
     PUNT         4103
     RDLABL       413B
     SYMSCN       4189
     ENDLOC       41A5
IN

*K, I8
IN

*N, FORTA7,,, B
IN
```

```
*K, I6, P8
IN

*P
    FTN        2991
    GOB        309D
    CNVT       30B5
    DUMMY      30F3
    FCMSTK     31D8
    GETSYM     3261
    IOPRBB     329A
    KCPART     346A
    KOUTPT     349B
    KPCSTK     34AD
    KPC3PR     3868
    KSYMGN     3880
    LABKPC     38C8
    LABLER     38DC
    PUNT       38FA
    Q8PRMS     3910
    STORE      3921
    SYMBOL     394F
    TSALOC     39F2
    LOCLB1     3A7D
    DUMYB1     3B10
    ARAYSZ     3B39
    ASSEM      3BA3
    BANANA     3C0A
    BGINDO     3CCD
    END        3DD6
    ENTCOD     3E1E
    HELEN      3EC7
    INXRST     401E
    NOPROC     4032
    PHASEB     4063
    READIR     44B8
    SUBFUN     4510
    SYMSCN     4577
    ENDLOC     4593
IN

*K, I8
IN

*N, FORTB1,,, B
IN

*K, I6, P8
IN
```

```
*P
     FTN          2991
     GOB          309D
     CNVT         30B5
     DUMMY        30F3
     FCMSTK       31D8
     GETSYM       3261
     IOPRBB       329A
     KCPART       346A
     KOUTPT       349B
     KPCSTK       34AD
     KPC3PR       3868
     KSYMGN       3880
     LABKPC       38C8
     LABLER       38DC
     PUNT         38FA
     Q8PRMS       3910
     STORE        3921
     SYMBOL       394F
     TSALOC       39F2
     LOCLB2       3A7D
     ACP          3B12
     AFIDL        3F58
     ASUPER       3FB2
     CGOTO        4068
     FINK         40C3
     INTRAM       4178
     PARTSB       4351
     SUBPR1       43F3
     SUBPR2       4431
     SUBPR3       44BE
     ENDLOC       4505
IN

*K, I8
IN

*N, FORTB2,,,B
IN

*K, I6, P8
IN

*P
     FTN          2991
     GOB          309D
     CNVT         30B5
     DUMMY        30F3
     FCMSTK       31D8
     GETSYM       3261
```

```
          IOPRBB       329A
          KCPART       346A
          KOUTPT       349B
          KPCSTK       34AD
          KPC3PR       3868
          KSYMGN       3880
          LABKPC       38C8
          LABLER       38DC
          PUNT         38FA
          Q8PRMS       3910
          STORE        3921
          SYMBOL       394F
          TSALOC       39F2
          LOCLB3       3A7D
          ACP          3B10
          ARITHR       3F56
          ASUPER       4114
          FINK         41CA
          INTRAM       427F
          PARTSB       4458
          SUBPR1       44FA
          SUBPR2       4538
          SUBPR3       4505
          ENDLOC       460C
IN

*K, I8
IN

*N, FORTB3, , , B
IN

*K, I6, P8
IN

*P
          FTN          2991
          GOC          3583
          BKDWN        3594
          BLDUP        35F3
          BSS          3636
          CHKWD        3654
          CHOP         37C8
          CL12         39DC
          CON          3A95
          COUNT        3ACC
          DATAST       3AE3
          GETSYM       3B8A
          INOUT        3C2E
          IXOPT        3C9D
```

```
     PHASEC         3DD7
     LABEL          416F
     LABIN          4191
     QXLD           41F7
     REED           4287
     SKIP           42E4
     SYMSCN         433A
     IOPRBC         4356
     Q8PRMS         45E0
     ENDLOC         45F1
IN

*K, I8
IN

*N, FORTC1,,, B
IN

*K, I6, P8
IN

*P
     FTN            2991
     GOOD           2E03
     INDEX          2E28
     IOPRBD         2E44
     NPUNCH         30F4
     Q8PRMS         3230
     PHASE6         3241
     LOCLD1         32E1
     DUMYD1         33A4
     AMT            33B1
     AMOUT          33BA
     ADMAX          39B3
     BKDWN          3BB1
     COUNT          3C1A
     LABOUT         3C31
     NP2OUT         3D10
     RBDX           3D3F
     RBPK           3D7B
     TABDEC         3DA5
     UNPUNC         3E29
     GETSYM         3E3F
     SYMSCN         3E7B
     ENDLOC         3E9D
IN

*K, I8
IN

*N, FORTD1,,, B
IN
```

```
*K, I6, P8
IN
*P
     FTN          2991
     GOOD         2E03
     INDEX        2E28
     IOPRBD       2E44
     NPUNCH       30F4
     Q8PRMS       3230
     PHASE6       3241
     LOCLD2       32E1
     DUMYD2       33A5
     AMT          33AC
     GETSYM       33B3
     IACON        33E1
     IHCON        3439
     NWRITE       3466
     PACK         34A1
     SYMSCN       34CC
     BEGINO       34E8
     FINISH       3694
     ENDLOC       3808
IN
*K, I8
IN
*N, FORTD2,,, B
IN
*K, I6, P8
IN
*P
     FTN          2991
     GOE          2E03
     INDEX        2E28
     IOPRBD       2E44
     NPUNCH       30F4
     Q8PRMS       3230
     PHASE6       3241
     LOCLD1       32E1
     DUMYD1       33A4
     AMT          33B1
     AMOUT        33BA
     ADMAX        398E
     BKDWN        3B8C
     COUNT        3BF5
     LABOUT       3C0C
     NP2OUT       3D1E
```

```
          RBDX          3D56
          RBPK          3D93
          TABDEC        3DBD
          UNPUNC        3E39
          CONV          3E4F
          GETSYM        3E88
          IACON         3ED5
          IHCON         3F2D
          NWRITE        3F59
          PACK          3F94
          SETPRT        3F94
          SYMSCN        4139
          ENDLOC        4155
IN

*K, I8
IN

*N, FORTE1,,, B
IN

*K, I6, P8
IN

*P
          FTN           2991
          GOE           2E03
          INDEX         2E28
          IOPRBD        2E44
          NPUNCH        30F4
          Q8PRMS        3230
          PHASE6        3241
          LOCLD2        32E1
          DUMYD2        33A5
          AMT           33AC
          CONV          33B3
          GETSYM        33EC
          IACON         3439
          IHCON         3491
          NWRITE        34BD
          PACK          34F8
          SETPRT        3523
          SYMSCN        3699
          BEGINO        36B5
          FINISH        37FE
          ENDLOC        396D
IN

*K, I8
IN
```

```
*N, FORTE2,,,B
IN

*K, I6, P8
IN

*L, FTN
IN

*L, Q8IFRM
IN

*L, Q8FS
IN

*L, Q8TRAN
IN

*L, FLOT
IN

*L, Q8QINI
IN

*L, Q8QEND
IN

*L, Q8CMP1
IN

*L, Q8RWBU
IN.

*L, Q8ERRM
IN

*L, Q8DFNF
IN

*L, Q8QX
IN

*L, Q8QUN1
IN

*L, Q8FGET
IN

*L, Q8MAGT
IN

*L, Q8QBCK
IN

*L, IOCK
IN
```

```
*L, Q8PSE
IN

*L, Q8PAND
IN

*L, Q8EXP9
IN

*L, Q8EXP1
IN

*L, Q8AB
IN

*L, SIGN
IN

*L, EXP
IN

*L, SQRT
IN

*L, ALOG
IN

*L, TANH
IN

*L, SIN
IN

*L, ATAN
IN

*L, QSAVE
IN

*L, IFALT
IN

*L, Q8FX
IN

*L, Q8PREP
IN

*U
```

Type: *Z

Press: CARRIAGE RETURN

Message: J

2.0B

```
    IN

    *K, I6, P8
    IN

    *P

         FTN            25EA
         GOA            2CAO
         CFIVOC         23E4
         CKNAME         2D42
         CNVT           2D52
         CONV           2D90
         DIAG           2DC3
         EXP9           2E53
         FLOAT          2F00
         GETC           304A
         GETF           3075
         GETSYM         336D
         GPUT           33A6
         IGETCF         33CF
         IOPRBA         33E8
         PACK           3684
         Q8PRMS         36A9
         RDLABL         36BA
         STORE          3708
         SYMBOL         3736
         ENDDO          37DD
         GNST           38DE
         OPTION         3A81
         OUTENT         3AC1
         PHASEA         3AF5
         PLABEL         3FD4
         STCHAR         402A
         TYPE           405C
         LOCLA1         4259
         DUMYA1         4316
         Q8QBDS         437D
         ENDLOC         437D
    IN

    *K, I8
    IN

    *N, FORTA1,,, B
    IN

    *K, I6, P8
    IN
```

```
*P
    FTN        25EA
    GOA        2CAO
    CFIVOC     2CE4
    CKNAME     2D42
    CNVT       2D52
    CONV       2D90
    DIAG       2DC3
    EXP9       2E53
    FLOAT      2F00
    GETC       304A
    GETF       3075
    GETSYM     336D
    GPUT       33A6
    IGETCF     33CF
    IOPRBA     33E8
    PACK       3684
    Q8PRMS     36A9
    RDLABL     36BA
    STORE      3708
    SYMBOL     3736
    ENDDO      37DD
    GNST       38DE
    OPTION     3A81
    OUTENT     3AC1
    PHASEA     3AF5
    PLABEL     3FD4
    STCHAR     402A
    TYPE       405C
    LOCLA2     4259
    DUMYA2     4316
    BYEQPR     437D
    CHECKF     456F
    COMNPR     460F
    CONSUB     46A5
    DATAPR     472C
    DIMPR      48BC
    EXRLPR     4A43
    FGETC      4AA1
    FORK       4AC0
    PEQVS      4C37
    PRNTNM     5016
    SUBPPR     50A3
    SYMSCN     5163
    TYPEPR     517F
    ENDLOC     5196
IN
```

```
*K, I8
IN

*N, FORTA2,,, B
IN

*K, I6, P8
IN

*P

     FTN          25EA
     GOA          2CA0
     CFIVOC       2CE4
     CKNAME       2D42
     CNVT         2D52
     CONV         2D90
     DIAG         2DC3
     EXP9         2E53
     FLOAT        2F00
     GETC         304A
     GETF         3075
     GETSYM       336D
     GPUT         33A6
     IGETCF       33CF
     IOPRBA       33E8
     PACK         3684
     Q8PRMS       36A9
     RDLABL       36BA
     STORE        3708
     SYMBOL       3736
     ENDDO        37DD
     GNST         38DE
     OPTION       3A81
     OUTENT       3AC1
     PHASEA       3AF5
     PLABEL       3FD4
     STCHAR       402A
     TYPE         405C
     LOCLA3       4259
     DUMYA3       4316
     ARAYSZ       437D
     ASEMPR       43E7
     ASGNPR       4591
     BDOPR        45D7
     CHECKF       4711
     CKIVC        47B1
     CONSUB       47C1
     CPLOOP       4848
     DATAPR       48ED
     FGETC       4A7D
     FORK        4A9C
     ERBPR       4C13
     MODMXR      4C66
     PUNT        50BD
     ENDLOC      50F5
IN

*K, I8
IN

*N, FORTA3,,, B
IN

*K, I6, P8
IN

*P

     FTN         25EA
     GOA         2CA0
     CFIVOC      2CEA
     CKNAME      2D42
     CNUT        2D52
     CONV        2D90
     DIAG        2DC3
     EXP9        2E53
     FLOAT       2F00
     GETC        304A
     GETF        3075
     GETSYM      336D
     GPUT        33A6
     IGETCF      33CF
     IOPRBA      33E8
     PACK        3684
     Q8PRMS      36A9
     RDLABL      36BA
     STORE       3708
     SYMBOL      3736
     ENDDO       37DD
     GNST        38DE
     OPTION      3A81
     OUTENT      3AC1
     PHASEA      3AF5
     PLABEL      3FD4
     STCHAR      402A
     TYPE        405C
     LOCLA4      4259
     DUMYA4      4316
     ARITH       437D
     SUBSCR      49EF
```

```
        TREE          4CAD
        ENDLOC        51A1
IN

*K, I8
IN

*N, FORTA4,,,B
IN

*K, I6, P8
IN

*P

        FTN           25EA
        GOA           2CA0
        CFIVOC        2CE4
        CKNAME        2D42
        CNVT          2D52
        CONV          2D90
        DIAG          2DC3
        EXP9          2E53
        FLOAT         2F00
        GETC          304A
        GETF          3075
        GETSYM        336D
        GPUT          33A6
        IGETCF        33CF
        IOPRBA        33E8
        PACK          3684
        Q8PRMS        36A9
        RDLABL        36BA
        STORE         3708
        SYMBOL        3736
        ENDDO         37DD
        GNST          38DE
        OPTION        3A81
        OUTENT        3AC1
        PHASEA        3AF5
        PLABEL        3FD4
        STCHAR        402A
        TYPE          405C
        LOCLA5        4259
        DUMYA5        4316
        BDOPR         437D
        CKIVC         44B7
        IOSPR         44C7
        ENDLOC        4B69
IN
```

```
*K, I8
IN

*N, FORTA5,,, B
IN

*K, I6, P8
IN

*P
     FTN          25EA
     GOB          2CFA
     CNVT         2DIO
     DUMMY        2D4E
     FCMSTK       2E33
     GETSYM       2EBC
     IOPRBB       2EF5
     KCPART       347D
     KOUTPT       34AE
     KPCSTK       34C0
     KPC3PR       387B
     KSYMGN       3893
     LABKPC       38DB
     LABLER       38EF
     PUNT         390D
     Q8PRMS       3923
     STORE        3934
     SYMBOL       3962
     TSALOC       3A05
     ARAYSZ       3A90
     ASSEM        3AFA
     BANANA       3B61
     BGINDO       3C24
     END          3D2D
     ENTCOD       3D75
     HELEN        3E1E
     INXRST       3F75
     NOPROC       3F89
     PHASEB       3FBA
     READIR       440F
     SUBFUN       4467
     SYMSCN       44CE
     ACP          44EA
     AFIDL        4930
     ASUPER       498A
     CGOTO        4A40
     FINK         4A9B
     INTRAM       4B50
     PARTSB       4D29
     SUBPR1       4DCB
```

```
    SUBPR2      4E09
    SUBPR3      4E96
    ARITHR      4EDD
    ENDLOC      509B
IN

*K, I8
IN

*N, FORTB1,,, B
IN

*K, I6, P8
IN

*P

    FTN         25EA
    GOC         31E0
    BKDWN       31F8
    BLDUP       3257
    BSS         329A
    CHKWD       32B8
    CHOP        342C
    CL12        3640
    CON         36F9
    COUNT       3730
    DATAST      3747
    GETSYM      37EE
    INOUT       3892
    IOPRBC      3901
    IXOPT       47AC
    LABEL       48E6
    LABIN       4908
    PHASEC      496E
    Q8PRMS      4CB8
    QXLD        4CC9
    REED        4D59
    SKIP        4DB6
    SYMSCN      4E0C
    ENDLOC
IN

*K, I8
IN

*N, FORTC1,,, B
IN

*K, I6, P8
IN
```

```
*P
     FTN          25EA
     GOOD         31E0
     AMOUT        3203
     ADMAX        37C8
     BEGINO       39C6
     BKDWN        3AD6
     COUNT        3B3F
     FINISH       3B56
     GETSYM       3CC5
     IACON        3D69
     IHCON        3DC1
     INDEX        3DEE
     IOPRBD       3E0A
     LABOUT       43B7
     NP2OUT       4496
     NPUNCH       44C5
     NWRITE       4601
     PACK         463C
     PHASE6       4667
     Q8PRMS       4703
     RBDX         4714
     RBPK         4750
     SYMSCN       477A
     TABDEC       4796
     UNPUNC       481A
     ENDLOC       4830
IN

*K, I8
IN

*N, FORTD1,,, B
IN

*K, I6, P8
IN

*P
     FTN          25EA
     GOE          31E0
     AMOUT        3203
     ADMAX        37D7
     BEGINO       39D5
     BKDWN        3B16
     CONV         3B7F
     COUNT        3BB8
     FINISH       3BCF
     GETSYM       3D3E
     IACON        3DE2
```

```
       IHCON         3E3A
       INDEX         3E66
       IOPRBD        3E82
       LABOUT        442F
       NP2OUT        4541
       NPUNCH        4579
       NWRITE        46B5
       PACK          46F0
       PHASE6        471B
       Q8PRMS        47B7
       RBDX          47C8
       RBPK          4805
       SETPRT        482F
       SYMSCN        49A9
       TABDEC        49C5
       UNPUNC        4A41
       ENDLOC        4A57
IN

*K, I8
IN

*N, FORTE1,,, B
IN

*K, I6, P8
IN

*L, FTN
IN

*L, Q8IFRM
IN

*L, Q8FS
IN

*L, Q8TRAN
IN

*L, FLOT
IN

*L, Q8QINI
IN

*L, Q8QEND
IN

*L, Q8CMP1
IN

*L, Q8RWBU
IN
```

```
*L, Q8ERRM
IN

*L, Q8DFNF
IN

*L, Q8QX
IN

*L, Q8QUN1
IN

*L, Q8FGET
IN

*L, Q8MAGT
IN

*L, Q8QBCK
IN

*L, IOCK
IN

*L, Q8PSE
IN

*L, Q8PAND
IN

*L, Q8EXP9
IN

*L, Q8EXP1
IN

*L, Q8AB
IN

*L, SIGN
IN

*L, EXP
IN

*L, SQRT
IN

*L, ALOG
IN

*L, TANH
IN

*L, SIN
IN
```

```
*L, ATAN
IN

*L, QSAVE
IN

*L, IFALT
IN

*L, Q8FX
IN

*L, Q8PREP
IN

*U
```

Type: *Z

Press: RETURN

Message: J

ADDITIONS AND MODIFICATIONS 3

3.1 MACRO ASSEMBLER 2.0 MODIFICATIONS

3.1.1 SYSTEM MODIFICATION EXAMPLE

The following steps outline the procedures for replacing a file such as PASS1. File name and tape numbers will differ for each system.

1. Reassemble and punch the relocatable information for all programs in the specific pass in binary form (in this case PASS1) so they can be absolutized on the disk. All parts of the pass, in this case PASS1 and PA1PR2, must be present.

2. Type: *LIBEDT

   Press: CARRIAGE RETURN

   Message: LIB
   IN

3. Mount the relocatable paper tape of PASS1 on the paper tape reader.

4. Press: MASTER CLEAR on the paper tape reader

5. Type: *K, I2, P8

   Press: CARRIAGE RETURN

   Message: IN

6. Type: *P, F

   Press: CARRIAGE RETURN

   Message: L, 02 FAILED 02
   ACTION

7. Mount the relocatable tape for PA1PR2 on the paper tape reader.

8. Press: MASTER CLEAR on the paper tape reader

9. Type: RP

   Press: CARRIAGE RETURN

   The paper tape is read.

   Message: L, 02 FAILED 02
   ACTION

10. Type: CU

    Press: CARRIAGE RETURN

    Message: IN

11. Type: *K, I8

    Press: CARRIAGE RETURN

    Message: IN

12. Type: *N, PASS1, , , B

    Press: CARRIAGE RETURN

    Message: IN

3.1.2 MODIFICATION OF LIBRARY MACROS EXAMPLE

Use the library macro preparation routine on paper tape 2 to change or add macro definitions to the library macros. Macro Assembler paper tape 2 contains this routine which must be loaded by the 1700 operating system loader.

Input to the program is source macro definitions. Paper tape 3 (system library macros) contains an ENDMAC statement at the end; but the user defined library macros source input tape(s) cannot contain the ENDMAC statement.

Example of library macro preparation:

1. Type: *P to load the operating system loader

   Press: CARRIAGE RETURN

   Message: J

2. Mount paper tape 2 (the relocatable binary tape of LIBMAC) on the paper tape reader.
3. Press: READY on the paper tape reader
4. Type: *L

   Press: CARRIAGE RETURN

   The paper tape is read.

   Message: L, 02 FAILED 02
   ACTION

5. Type: CU

   Press: CARRIAGE RETURN

   Message: J

6. Mount the paper source tape of user defined macros on the paper tape reader.
7. Press: READY on the paper tape reader
8. Type: *X

   Press: CARRIAGE RETURN

   Message: L, 02 FAILED 02
   ACTION

9. Mount paper source tape 3 (system library macros) on the paper tape reader.

10. Press: READY on the paper tape reader

11. Type RP

    Press: CARRIAGE RETURN

    Paper tape 3 is read.

    The library macro skeleton permanent file is punched.

    Message: MACSKL END

12. Remove the new paper tape, NEW MACSKL, from the paper tape punch.

13. Press CARRIAGE RETURN

    The library macro directory file is punched.

    Message: J

To insert in the porgram library the library macro directory tape which was punched in step 13 and also the NEW MACSKL tape (the library macro skeleton permanent file) which was punched in step 11, use the following steps:

14. Type: *LIBEDT

    Press: CARRIAGE RETURN

    Message: LIB

15. Mount paper tape NEW MACSKL which was punched in step 11 on the paper tape reader.

16. Press: READY on the paper tape reader

17. Type: *N, MACSKL, , , B

    Press: CARRIAGE RETURN

    The paper tape NEW MACSKL is read.

    Message: L, 02 FAILED 02
    ACTION

18. Type: CU

    Press: CARRIAGE RETURN

    Message: IN

19. Mount the library macro directory paper tape punched in step 13 on the paper tape reader.

20. Press: READY on the paper tape reader

21. Type: *N, MACROS, , , B

    Press: CARRIAGE RETURN

    The library directory paper tape is read.

    Message: L, 02 FAILED 02
    ACTION

22. Type: CU
Press: CARRIAGE RETURN
Message: IN

23. Type: *Z
Press: CARRIAGE RETURN
Message: J

3.2 COSY 1.0 MODIFICATIONS

COSY allows 8 output devices within a single job. To modify the number of COSY output devices, reassemble with the following card changes:

With x as the number of output devices to be used, the TABSIZ card should read

```
                 DEL/          145
TABSIZ           NUM           x
```

TABLE is a BSS block of 8 words. Delete or add words to allow only enough words for each device to be used. x is the number of output devices to be used.

```
                 DEL/          595
TABLE            BSS           TABLE(x)
```

3.3 MASS STORAGE FORTRAN 2.0A AND 2.0B MODIFICATIONS

3.3.1 LOADING AND CALLING SELCOP

SELCOP is a utility program helpful in building a 1700 FORTRAN installation tape or deck. It consists of two programs: SELCOP and IOCAL. IOCAL handles the I/O for SELCOP.

The SELCOP program allows an operator to build a tape from either a tape or a deck of relocatable binary programs. SELCOP may:

select a binary relocatable program from the equipped input unit and copy the program on the equipped output unit

change equipped units during program execution

rewind any tape drive

transfer a number of records from one unit to another without using the system standard units

Loading SELCOP and IOCAL into Program Library

1. MSOS 2.1 must be installed.

2. Type: *P

   Press: CARRIAGE RETURN

   Message: J

3. Load the SELCOP paper tape into the paper tape reader.

   Press: CLEAR on the reader

4. Type: *K, I2

   Press: CARRIAGE RETURN

   Message: J

5. Type: *LIBEDT

   Press: CARRIAGE RETURN

   Message: LIB
   IN

6. Type: *L, SELCOP

   Press: CARRIAGE RETURN

   Part of the paper tape is read.

   Message: IN

7. Type: *L, IOCAL

   Press: CARRIAGE RETURN

   The rest of paper tape is read.

   Message: IN

Calling SELCOP

To call SELCOP, the object library must be installed.

1. Type: *P

   Press: CARRIAGE RETURN

2. Type: SELCOP

   Press: CARRIAGE RETURN

   Message: IN

3. Type: one of the five commands listed under SELCOP commands

4. Message: NEXT

5. Type: another of the five commands if desired

6. Since SELCOP is written in 1700 FORTRAN, any errors of incorrect input of logical unit formats will result in a FORTRAN I/O ERROR, and any errors in number of logical units will result in a 1700 MSOS J02 error. When the program terminates because of errors, recall SELCOP and check correct formats and logical unit numbers for the installation.

SELCOP Commands

Equipping Command (*K): The *K statement must precede the *N command. Its function is to equip the input unit and the punch unit. This command only affects SELCOP; it does not affect MSOS.

Type: *K

Press: CARRIAGE RETURN

Type: lu, lu
lu, lu are the 2 two-digit parameters of logical units to be equipped.

Example:

02, 03 equips the paper tape reader as input, paper tape punch as output.

Transfer Command (*T): The *T statement is used to transfer a number of records from one unit to another. The I/O consists of formatted binary reads and writes.

Type: *T

Press: CARRIAGE RETURN

Type: lu, lu, xxxx
lu, lu are the 2 two digit parameters of logical units to be equipped.

xxxx is a four-digit record count.

Since the reads from input comments are in FORTRAN, it is important to use 2-digit logical unit numbers and 4-digit record counts.

Example:

02, 06, 0030 tells SELCOP to transfer thirty records from logical unit 2 to logical unit 6.

Name Command (*N): The *N is the main SELCOP command. When SELCOP receives a six-character name, it searches the input unit which was equipped by the *K statement to find a 2050_{16} NAM block. If that NAM block is the name which was input from the teletypewriter, SELCOP copies the program on output unit until it finds a $C050_{16}$ XFR block which terminates the command.

Type: *N

Press: CARRIAGE RETURN

Type: a 6-character ASCII name (if the name is fewer than 6 characters, right fill to 6 characters with blanks).

Example:

VERIFY will find program VERIFY on the input unit and copy it onto the output unit.

Rewind Command (*R): Use the *R statement to rewind a specific magnetic tape drive.

Type: *R

Press: CARRIAGE RETURN

Type: one two-digit logical unit number

Example:

Typing 07 rewinds tape drive 1 which is logical unit 7.

Stop Command (*S):

Type: *S

Press: CARRIAGE RETURN

Message: J (this signifies entry into the system; SELCOP terminates)

3.3.2 BUILDING A MASS STORAGE FORTRAN 2.0A OR 2.0B INSTALLATION TAPE

Requirements

To build a Mass Storage FORTRAN installation tape, the following requirements must first be met:

Installation of MSOS 2.1

Installation of Macro Assembler

Installation of SELCOP

Installation of FORTRAN 2.0A or 2.0B

20K memory for FORTRAN 2.0A; 24K for 2.0B

Either:

- 2 magnetic tape units, or
- 1 magnetic tape unit; 1 paper tape reader; and 1 paper tape punch

Procedures

1. Compile or assemble all of the source tape programs using the P option to punch on either magnetic or paper tape. Sections 4.3.4 (2.0A) and 4.3.8 (2.0B) list installation tape contents.
2. Call SELCOP using the procedures outlined in 3.3.1.
3. Use SELCOP's *T statement as described in 3.3.1 to transfer the control characters from the teletype to the tape being constructed.

Type:	*T
Press:	CARRIAGE RETRUN
Type:	lu, lu, 0002
Press:	CARRIAGE RETURN

For magnetic tape:

Type:	*K, I6, P8
Press:	LINE FEED
Press:	CARRIAGE RETURN
Type:	*P
Press:	CARRIAGE RETURN

For paper tape:

Type:	*K, I2, P8
Press:	LINE FEED
Press:	CARRIAGE RETURN
Type:	*P
Press:	CARRIAGE RETURN

4. Use the *N statement (3.3.1) to select and copy the programs from the binaries generated in step 1 onto the tape being generated, beginning with FTN and continuing in the order given in the tape contents section as: GOA, CNVT, ..., ENDLOC. Note that the logical units must be equipped with the *K statement before the *N statement is used.

5. Use the *T statement (3.3.1) to transfer the control characters. Then:

Type:	*T
Press:	CARRIAGE RETURN
Type:	lu, lu, 0003
Press:	CARRIAGE RETURN

Type:	*T
Press:	LINE FEED
Press:	CARRIAGE RETURN
Type:	*K, I8
Press:	LINE FEED
Press:	CARRIAGE RETURN
Type:	*N, FORTxx, , , B xx is the last record in the phase being built
Press:	LINE FEED
Press:	CARRIAGE RETURN

6. If paper tape is being used, it is suggested that the following be typed at the end of each phase:

Type:	*U
Press:	LINE FEED
Press:	CARRIAGE RETURN

7. Repeat steps 2 through 6 for each phase.

3.3.3 CONSTRUCTION OF OBJECT LIBRARY

Use the following steps for each object library:

1. Call SELCOP as outlined in 3.3.1.
2. Use the *T statement (3.3.1) to transfer the control character from the teletype to the tape being constructed.

Type:	*T
Press:	CARRIAGE RETURN
Type:	lu, lu, 0001
Press:	CARRIAGE RETURN
Type:	*L, entry point name Part 4.3.7 lists entry point names for 2.0A Part 4.3.11 lists entry point names for 2.0B
Press:	CARRIAGE RETURN

3. Use the *N statement (3.3.1) to select and copy the program from the tape of binaries onto the tape being generated.

4. Repeat steps 2 and 3 for each specified entry point.

5. After entering the last program in the program library, transfer an *U onto the tape being generated.

Type:	*T
Press:	CARRIAGE RETURN
Type:	lu, lu, 0001
Press:	CARRIAGE RETURN
Type:	*U
Press:	CARRIAGE RETURN

3.3.4 PHASE MODIFICATION

1. Compile and/or assemble all programs which appear in the phase to be modified. The relocatable output must be put in absolute form according to the order specified in the FORTRAN 2.0A or 2.0B tape formats in part 4.3.5 or part 4.3.9 to establish a relocatable tape of the programs in the modified phase.

2. Type: *P

Press:	CARRIAGE RETURN
Message:	J

3. Type: *LIBEDT

Press:	CARRIAGE RETURN
Message:	LIB IN

4. If input is from paper tape:

   Type: *K, I2, P8

   Press: CARRIAGE RETURN

   If input is from magnetic tape:

   Type: *K, I6, P8

   Press: CARRIAGE RETURN

5. Type: *P

   Press: CARRIAGE RETURN

   Action: The system reads the tape

   Message: IN (if there is a *T at the end of the tape)

6. Type: *K, I8

   Press: CARRIAGE RETURN

   Message: IN

7. Type: *N, file name of modified phase,,, B

   Part 4.3.5 lists FORTRAN 2.0A phase names
   Part 4.3.9 lists 2.0B phase names

   Press: CARRIAGE RETURN

   Message: IN

3.3.5 OBJECT LIBRARY MODIFICATION

1. When a subroutine in the object-time library needs to be changed, assemble or compile the routine on a relocatable tape.

2. Type: *P

   Press: CARRIAGE RETURN

   Message: J

3. Type: *LIBEDT

   Press: CARRIAGE RETURN

   Message: LIB
   IN

4. Type: *L, routine entry point name

   Entry point names are in 4.3.7. They are the same for 2.0A and 2.0B

   Press: CARRIAGE RETURN

   Message: IN

3.4 RE-ENTRANT FORTRAN LIBRARY PACKAGE

If the RDISP module is used with the Re-entrant FORTRAN Library Package, FMASK and FLIST may require modification. If RDISP is not used, FMASK and FLIST may be removed from SYSBUF.

FMASK is a location which indicates the software priority levels requiring the saving of the temporary area used by the FORTRAN routines. Do not assign these levels to interrupt lines, since the interrupt handler does not save the FORTRAN data. Set to one each bit position in FMASK that corresponds to each level using FORTRAN. If too many levels are allowed to run FORTRAN programs, the overhead for the low-priority programs may be unnecessarily high.

Example:

```
     FMASK        NUM       $008C
```

This allows FORTRAN at levels 2, 3, and 7.

Levels 0 and 1 are reserved for unprotected programs and do not interrupt higher priority levels using FORTRAN. Therefore, the mask is not set for levels 0 and 1.

Table FLIST is the table of entry point locations in the FORTRAN library which must be saved to allow re-entrant use of the library. The symbolic names must also be declared as externals (EXT) and must appear as entry names (ENT) in the library subroutines.

```
     FLIST        ADC       FEND-*-1
                  ADC       Q8Q12F, Q8QF2I, Q8AB, Q8SG, EXP, SORT, ALOG
                  EXT       Q8Q12F, Q8QF2I, Q8AB, Q8SG, EXP, SORT, ALOG
                  EQU       FEND(*)
```

An example of SYSBUF modified for the re-entrant FORTRAN library package is contained on the COSY tape under the deck name SYSBFD. This may be used or the user may modify SYSBUF for his own needs.

Installation procedures are:

1. Assemble RDISP and obtain a relocatable binary paper tape
2. Decompress and assemble SYSBFD or an equivalent and obtain a relocatable binary paper tape
3. Delete DISP and SCHEDU and replace with RDISP. Replace SYSBUF with the revised version of SYSBUF in the installation tape
4. Install the system as it would normally be installed.

The re-entrant FORTRAN library is necessary if more than one priority level is written in FTN. It requires a special version of Scheduler (SCHEDU) and Dispatcher (NDISP) for which the RDISP is substituted.

3.5 OUTPUT MESSAGE BUFFERING PACKAGE

3.5.1 REQUIREMENTS

Reserve buffer area in core or in mass memory for exclusive use of the buffer package at system initialization time. Reserve at least three times the maximum record size.

The user may reserve the last 1000 sectors of the disk for the software buffering package. SECTOR is set to 2FFF during initialization, and the remaining sectors are used.

For core buffering, put a BSS block in SYSBUF.

For mass memory buffers, a *M, hhhh, s initializer statement may also be used.

A word addressable disk or drum driver is necessary for this version.

Replace the normal version of SYSBUF with one which defines the physical device tables for the software buffered devices.

Include BUFFER as a core resident program.

3.5.2 INSTALLATION PROCEDURES

1. The output message buffering package is added to the 1700 Operating System in the same way as is a driver. Insert a buffer table and a character buffer area for each buffer input logical unit by using the BUFFER macro. The BUFFER macro generates a physical device table for each buffered device.

   BUFFER f, l, h, lu, rp, n

   f start address of buffer

   l end of buffer address plus 1

   h most significant bits of mass memory buffer word address; to be blank for core buffer

   lu logical unit for actual output

   pr request priority for buffer output on this logical unit

   n character buffer size for actual output

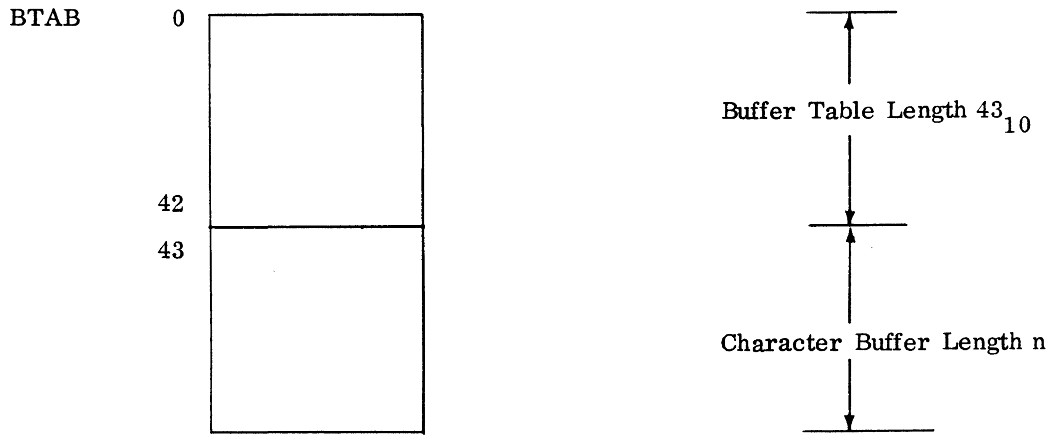
   When using the BUFFER macro, define the internal symbols as in the following example:

   ```
               EQU            BFLEVL(10)
               EQU            BFMMLU($8C2)
   ```

   BFLEVL is the priority level of the buffer package

   BFMMLU is the mass memory device logical unit

   The BUFFER macro generates the following:



n is the length of the character buffer specified in the macro call

BTAB is the address of the buffer table that must be put in the LOG1A logical unit table

The first 13 words of the table correspond to the standard 13 words required for all the physical equipment tables for all devices. The additional parameters define buffering parameters, the available core or mass memory area, and the character buffer size. The character buffer follows the last word of the buffer table.

2. Add a buffer table address to LOG1A:

```
ADC        BTAB1
```

3. Add to LOG1

```
ADC        0
```

4. Add to LOG2:

```
ADC        $FFFF
```

5. Repeat steps 1 through 4 for each buffer input logical unit.

6. Assemble SYSBUF with all the revised tables, as desired, or use the example provided on the COSY tape under the deckname SYSBFB. This example provides the following buffered output devices using mass memory buffers which occupy an area from sector 2FFF on:

   Teletypewriter

   1728 card punch

   1723 paper tape punch

   1742 printer

7. Assemble the BUFFER module and obtain a relocatable paper tape.

8. Insert BUFFER and replace the existing SYSBUF with SYSBUF (revised version) on the installation tape as core resident programs.

9. Install the system as it would normally be installed.

OUTPUT BUFFERING PACKAGE PHYSTB GENERATED BY BUFFERED MACRO

0	$1200+LV		ELVL	STANDARD PHYSTB
1	BUF DRI	Initiator Entry	EDIN	
2	BUFDRC	Continuator Entry	EDCN	
3	BUFDRC	Diagnostic Entry	EDPGM	
4	-1	Diagnostic Clock	EDCLK	
5	0	Logical Unit Assigned	ELU	
6	0	Request Address	EPTR	
7	0	Hardware Address	EWES	
8	$A4	Type Code	EREQST	
9	0	Status Word 1	ESTAT1	
10	0	Start Core Address	ECCOR	
11	0	End Core Address +1	ELSTWD	
12	0	Status Word 2	ESTAT2	
13	0	Number of Attempts	TIMER	
14	F	Buffer Start	LOCB	
15	L	Buffer End +1	ENDB	
16	F	Temporary Buffer Start	FIRST	
17	L	Temporary Buffer End +1	LAST	
18	$04F 0+LV	Mass Memory WRITE	DPL0	
19	BWRITC	Completion Address	1	STANDARD PHYSTB
20	0	Thread	2	
21	BFMMLU	Logical Unit	3	
22	0	Length	DLEG	
23	0	Core Address	DART	
24	H	MSB of Mass Memory Address	DTRACK	
25	F	Buffer Store Pointer	STOR	

OUTPUT BUFFERING PACKAGE PHYSTB GENERATED BY BUFFERED MACRO (contd)

26	0	Control Word	CONTRL	
27	$200+16*LV+LV	Mass Memory READ	DOUT0	
28	BREADC	Completion Address	1	STANDARD PHYSTB
29	0	Thread	2	
30	BFMMLU	Logical Unit	3	
31	0	Length	OUTLNG	
32	CHBUFF	Core Address	DADR	
33	H	MSB of Mass Memory Address	OUTTK	
34	F	Buffer Read Point	READ	
35	0	Control Word	SKELNG	
36	$C00+16*RP+LV	Character Output FWRITE	OUTP0	
37	BOUTPC	Completion Address	1	STANDARD PHYSTB
38	0	Thread	2	
39	LU	Output Logical Unit	3	
40	0	Length	4	
41	CHBUFF	Address of Character Buffer	ACHAR	
42	N	Length of Character Buffer	LCHAR	

3.6 DRIVER ADDITION

To insert a driver, the following core resident modules must be modified or added:

Logical unit tables LOG1A, LOG1, LOG2	Section III.1.2.2
Interrupt mask table (MASKT)	Section III.1.2.3
Diagnostic timer table (DGNTAB)	Section III.1.2.9
Physical device table (PHYSTB)	Section III.1.2.12
Interrupt response routines	Section III.1.2.13
Interrupt trap area	Section III.1.1.3

Each step which is unique to each driver is outlined in this section. For additional information on I/O Modification, such as table structures, see sections in Part III as listed above.

Driver deletion is described in step 5 of II. 2.

3.6.1 STANDARD INSTALLATION I/O CAPABILITIES

The drivers incorporated on the standard install tape allow formatted read and writes and/or unformatted read and writes on all devices except on magnetic tape. The magnetic tape driver only allows formatted I/O. To have unformatted I/O for magnetic tape, add RWBA and delete *S, RWBA, 7FFF from the install tape.

3.6.2 1573 TIMER

Install the timer (TIMINT) routine at system initialization time.

1. Rebuild the installation tape to include the TIMINT routine. Before assembling the system tables, initialize the following timer external parameters which define system variables and are necessary for timer operation:

   TIMCPS determines timer operating frequency. If, for example, TIMCPS is equated to 60 (as in the example below) the timer interrupts every 60 seconds.

   TIMACK contains equipment and station used to acknowledge timer interrupt. Modify this parameter if the equipment number for the 1750 data and Control Terminal is also modified.

   NSR establishes upper limit on the number of timer completion addresses scheduled for each timer interrupt. Excess addresses are handled on the next interrupt.

2. Insert the following coding sequence in the System Tables:

LABEL	OP	ADDRESS	COMMENTS
	ENT	TODLVL	TIME OF DAY TIMER REQ. WITH LEVEL
	ENT	TIMCPS	TIMER CYCLES PER SECOND
	ENT	TIMEC	TIMER CYCLES PER 1 SEC -1
	ENT	TIMACK	TIMER ACKNOWLEDGE CODE
	ENT	NSCHED	MAX. NUM. OF COMP ADRS PER INT
TIMCPS	EQU	TIMCPS(60)	
TIMEC	EQU	TIMEC (TIMCPS/10-1)	
TIMACK	EQU	TIMACK ($401)	

```
LABEL         OP        ADDRESS                  COMMENTS

NSR           EQU       NSR(5)
NSCHED        ADC       NSR
TODLVL        EQU       TODLVL($1, 006)
```

3. Initialize interrupt line x in LOCORE so that it will accommodate the timer interrupt. Insert the following coding in the interrupt trap area of LOCORE:

```
LABEL         OP        ADDRESS                  COMMENTS

LINEx         NUM       0
              RTJ       ($FE)              COMMON INTERRUPT HANDLER
              NUM       13                 PRIORITY LEVEL OF TIMER INT
              ADC       TIMINT             1573 TIMER INTERRUPT PROCESSOR
```

3.6.3 1711/1712/1713 TELETYPEWRITER DRIVER

Description

The 1711/1712/1713 teletypewriter driver executes under the CONTROL DATA 1700 Operating System to provide the capability for data input/output between core memory and the teletypewriter keyboard. The teletypewriter connects directly to the 1704 Computer and is part of the low-speed I/O Common Synchronizer Package.

The 1711/1712/1713 driver processes requests made by user programs for data transfer between core memory and the teletypewriter. The requests are READ, FREAD, WRITE, and FWRITE.

Installation Requirements

Core Memory: The following core memory is necessary.

Driver	319 words
Logical unit tables	3 words
Diagnostic timer table	1 word
PHYSTB	16 words
	339 words

Mass Memory: None.

Installation Procedures

The equipment code is preset to one for all low-speed I/O common synchronizer devices.

1. The following four-word interrupt entry must be in the interrupt trap area of the LOCORE program. It is associated with the low-speed I/O common synchronizer package and is assigned to interrupt LINE1:

```
LINE1           NUM         0
                RTJ-        ($FE)
                NUM         10
                ADC         EPROC
```

2. Enter the following into LOG1A:

```
                ADC         TELPTR
```

3. Enter the following into LOG1:

```
                ADC         0
```

4. Enter the following into LOG2:

```
                ADC         $FFFF
```

5. Declare the following entry point:

```
                ENT         TELPTR
```

6. Declare the following external:

```
                EXT         TYPEI, TYPEDR, TYPERR
```

7. In forming the PHYSTB for the 1711/1712/1713 teletypewriter driver, use the following information:

driver priority level	10
equipment type	0
equipment class	6

Add the PHYSTB to the system tables and parameters (SYSBUF) using the following coding:

WORD	LABEL	OP	ADDRESS
0	TELPTR	NUM	$120A
1		ADC	TYPEI
2		ADC	TYPEDR
3		ADC	TYPERR
4-6		NUM	-1,0,0
7-8		ADC	$91,$3006
9-15		BZS	(7)

8. If time-out surveillance over teletypewriter operation is desired, enter into the diagnostic timer table in SYSBUF the following entry:

```
                                        ADC          TELPTR
```

9. Modify MASKT according to instructions in Part III, Section 1.2.3.

3.6.4 1713 TELETYPEWRITER READER/PUNCH DRIVER

Description

The 1713 teletypewriter reader/punch driver provides either keyboard, or paper tape and printer, or paper tape output. The reader and punch modules reside on mass memory. The keyboard module must be core resident. It processes requests made by user programs between core memory and the teletypewriter.

Installation Requirements

Core Memory:

MASDRV and BUFFER (EQUATED TO LNGTH)	108 + buffer size
system tables and parameters	12
physical equipment tables	57
common continuator (S13CON)	40
diagnostic timer table	3
keyboard printer	370
system directory entries	14
	594 + buffer size

Mass Memory:

DRVMAC and reader	293
DRVMAC and punch	283
	576

Procedures

The following procedures are unique to the 1713 teletypewriter reader/punch driver.

1. The 1713 is part of the low-speed package which is loaded into the computer on interrupt line 1. Only the keyboard device table address must be included with the other device addresses using line 1. The 1713 reader and punch may be independent of line 1, since the continuator of the modules determines which module is active. Therefore, the reader and punch may be assigned any logical unit numbers. The 1713 reader and punch should not be assigned to any other interrupt line. Refer to Part III, 1.2.2 for information on interrupt line assignment.

```
LINE1       NUM     0
            RTJ-    ($FE)
            NUM     10
            ADC     EPROC
```

2. Insert the following into the LOG1A table:

```
            ADC     S13KBD             ENTRY IN KEYBOARD PHYSTB
            ADC     S13RDR             ENTRY IN READER PHYSTB
            ADC     S13PCH             ENTRY IN PUNCH PHYSTB
```

3. Insert the following into LOG1:

```
            ADC     0
            ADC     0
            ADC     0
```

4. Insert the following into LOG2:

```
            NUM     $FFFF
            NUM     $FFFF
            NUM     $FFFF
```

5. Insert the following into SECPRO for each module:

```
NUM          $7FFF
```

6. If the timer package is to be used, add the device table addresses to the diagnostic timer table.

```
ADC          S13KBD
ADC          S13RDR
ADC          S13PCH
```

7. Declare the following:

```
ENT          S13BZY, S13MOD
ENT          S13KBD, S13RDR, S13PCH
ENT          S13CON
EXT          S13KI, S13KC, S13KER
EXT          MASDRV, MI
EXT          M1713R, M1713P
```

8. Insert the following physical equipment tables:

WORD	LABEL	OP	ADDRESS	COMMENTS
	S13BZY	NUM	0	
	S13MOD	NUM	0	
*KEYBOARD PHYSTB				
		ADC	S13KC	KEYBOARD CONTINUATOR
		ADC	S13LI	KEYBOARD INITIATOR
0	S13KBD	NUM	$120A	
1		ADC	MASDRV	
2		ADC	S13CON	
3		ADC	S13KER	
4-8		NUM	-1, 0, 0, $91, $3266	
9-15		NUM	0, 0, 0, 0, 0, 0, 0	
*READER PHYSTB				
	S13RC	ADC	0	DRVMAC INSERTS ADDR. OF DRIVER CONT.
		ADC	M1713R	INDEX TO SYS. DIRECTORY

```
WORD          LABEL     OP      ADDRESS                COMMENTS

0             S13RDR    NUM     $120A
1                       ADC     MASDRV
2                       ADC     S13CON
3                       ADC     0                      DRVMAC INSERTS ADDR.
                                                       OF S13RER HERE
4-8                     NUM     -1, 0, 0, $91, $2282
9-16                    NUM     0, 0, 0, 0, 0, 0, 0, 0
              S13PC     ADC     0                      DRVMAC INSERTS ADDR OF
                                                       DRIVER CONT HERE
                        ADC     M1713P                 INDEX TO SYS DIRECTORY
*PUNCH PHYSTB
0             S13PCH    NUM     $120A
1                       ADC     MASDRV
2                       ADC     S13CON
3                       ADC     0                      DRVMAC INSERTS ADDR OF
                                                       S13PER HERE
4-8                     NUM     -1, 0, 0, $91, $2274
9-15                    NUM     0, 0, 0, 0, 0, 0, 0
```

The following is the common continuator which resides in system tables.

```
16            S13CON    LDQ*    S13BZY
17                      SQN     SA                     SKIP IF ANY UNIT BUSY
18                      JMP*    SE+1
19            SA        INQ     -KBDLU                 CHECK IF KBD INTERRUPTE
20                      SQN     SB                     SKIP IF NO
21                      JMP*    SE+1
22            SB        LDQ     =N$91
23                      ENA     0
24                      INP     SE-*
25                      STA*    SBA+1                  STORE STATUS
26                      AND     =N$38
27                      SAN     1
28                      JMP*    SB+2
```

WORD	LABEL	OP	ADDRESS	COMMENTS
29	SBA	LDA	=NO	
30		ALS	4	CHECK IF MI
31		LDQ*	S13BZY	
32		SAM	SC	SKIP IF MI
33		JMP*	SF	
34	SC	LDQ	LOG1A, Q	
35		LDQ-	(ZERO), Q	STORE FIRST WORD OF PDT IN REQ
36		STQ*	REQ	
37		RTJ-	($F4)	
38	REQ	NUM	$1200	SCHEDULE MAN INT PROCESSOR
39		ADC	MI	
40		LDQ*	S13BZY	
41		JMP*	SF	
42	SE	NOP	0	
43		ENQ	KBDLU	
44	SF	LDQ	LOG1A, Q	
45		TRQ	A	
46		INQ	-2	CONTINUATOR 2 LOCATIONS BEFORE PHYSTB
47		LDQ-	(ZERO), Q	
48		STQ-	I	
49		TRA	Q	
50		JMP-	(ZERO), I	GO TO CONT. IN DRIVER

9. Equate the logical unit of the keyboard to KBDLU as:

```
                    EQU          KBDLU(4)
```

There are two mass memory modules. One consists of the reader driver body S13002 with the macro DRVMAC; the other consists of the punch driver body S13003 with the macro DRVMAC.

The macro DRVMAC is a subprogram for each of these driver bodies. DRVMAC's function is to store the addresses of the initiator, the continuator, and the error routine in the particular mass memory driver's PHYSTB.

Each of the mass memory drivers will be released as a module containing the driver body and DRVMAC.

The following listing is that of a mass memory reader module. Within it, the macro call FRONT parameters are entry points in the driver body and are not the same for the reader and the punch. This example only uses the entry points in the reader body.

```
         NAM       DRVMAC
FRONT    MAC       P1, P2, P3
         EXT       'P1','P2','P3'
START    INA       LEN
         STQ-      I                STORE P.D.T. ADDRESS IN I.
         LDQ       =X'P3'           RELATIVE ADDR OF DRIVER ERROR
                                    ROUTINE
         AAQ       Q                ADD TO ADDR. DRIVER IS LOADED AT
         STQ-      3,I              STORE IN PDT
         LDQ-      I                DECREMENT I 2 TO STORE CONTINUATOF
         INQ       -2
         STQ-      I
         LDQ       -X'P2'           REL. ADDR. OF DRIVER CONT.
         AAQ       Q                TELETYPE READER AND PUNCH
                                    CONTINUATORS

         STQ-      (ZERO),I         ARE TWO LOCATIONS BEFORE PHSTAB.
         RAO-      I                SET I BACK TO PHSTAB ADDRESS.
         RAO-      I
         LDQ       =X'P1''          REL ADDR OF DRIVER INITIATOR
         AAQ       A
         LDQ-      I                SAVE PDT ADDR IN Q
         STA-      1,I
         STA-      I
         JMP-      (ZERO),I         JUMP TO INITIATOR
         EQU       ZERO($22)
         EQU       LEN(*-START)
         EMC
         FRONT     MS13RI, MS13RC, MS13RE
         END
```

10. Names which are associated with reader and punch ordinals are:

reader module:	*YM, M1713R, ordinal
punch module:	*YM, M1713P, ordinal

11. Load the punch and the reader modules under separate *M statements according to their ordinal. The LIBEDT *S statement is not needed to set request priority for these two modules.

```
*M     1713    READER
 DRVMAC     address
 S13002     address

*M     1713    PUNCH
 DRVMAC     address
 S13003     address
```

12. Install the keyboard module, MASDRV, as a core resident module under a *L statement.

    a. Equate the length of the largest driver module (including the length of DRVMAC) on mass memory using MASDRV:

    ```
    EQU          LNGTH(       )
    ```

    b. Equate the number of drivers on mass memory plus 1:

    ```
    EQU          NMASDR(       )
    ```

    c. When installing the standard release of the 1713, equate LNGTH to a value of 293_{10} and NMASDR to a value of 3.

13. Delete the statement *S, MAS300, 7FFF from the standard release installation tape.

    The following is an example of information which may appear on the teletypewriter as the 1713 is installed:

```
*S, ONE, 7FFF
*S, TWO, 7FFF
*S, THREE, 7FFF
*YM, LOADSD, 1, JOBENT, 2, JOBPRO, 3, JPLOAD, 4, JPST, 5
*YM, JPCHGE, 6, JBKILL, 7, JPT13, 8, MIPRO, 9, LIBEDT, 10
*YM, MOD1, 11, MOD2, 12, MOD3, 13, MOD4, 14, RESTOR, 15
*YM, ODEBUG, 16, RCOVER, 17, BRKPT, 18
*YM, M1713R, 19
*YM, M1713P, 20
*L         LOCORE
  LOCORE         0000
  SYSBUF         01EC
  SCHEDU         0681
  NDISP          0750
  NCMPRQ         078C
  NFNR           07BD
  ADEV           0827
*M         LOADER
  LOAD           0001
  BRANCH         0001
  LIDRIV         0001
  LCDRIV         0001
```

```
  LMDRIV          0001
  LLDRIV          0001
  SCAN            0001
  CHPU            0001
  ADJOVE          0001
  CONVRT          0001
  TABSCH          0001
  TABSTR          0001
  LSTOUT          0001
  LINK1           0001
  LINK2           0001
  COREXT          0001
  DPRADD          0001
  LOADER          0001
  NAMPRO          0001
  RBDBZS          0001
  ENTEXT          0001
  XFRPRO          0001
  HEXPRO          0001
  EOLPRO          0001
  ADRPRO          0001
*L          DRCORE
  DRCORE          0A2A
  ALCORE          0B5F
  ALVOL           0C08
  OFVOL           0C29
  TRVEC           0C35
  PARAME          0C52
  COMMON          0CB0
  NIPROC          0CC7
  NEPROC          0D42
  NMONI           0DA6
  RW              0DE8
  MAKQ            0E56
  MINT            0E79
*M          JOBENT
  JOBENT          0021
  T11             0021
  T7              0021
  T3              0021
*M          JOBPRO
  JOBPRO          0025
  PROTEC          0025
  T5              0025
*M          JPLOAD
  JPLOAD          0033
*M          JPST
  JPST            0038
```

```
*M          JPCHGE
 JPCHGE          003A
 ASCHEX          003A
*M          JBKILL
 JBKILL          003D
*M          JPT13
 JPT13           003F
 T13             003F
*M          MIPRO
 MIPRO           0045
*M          LIBEDT
 LIBEDT          0048
*M          UTILIB
 UTILIB          0053
*M          PLINSN
 PLINSN          0060
*M          FILE
 FILE            006D
*M          GENLIB
 GENLIB          007B
*M          RESTORE DEVICE
 RESTOR          0080
*M          ODEBUG
 ODEBUG          0083
*M          RCOVER
 RCOVER          0098
 OUTSEL          0098
 DMPCOR          0098
 MASDMP          0098
*M          BRKPT
 BRKPTD          00A1
 SIFT            00A1
 BIASCI          00A1
 RETJMP          00A1
 JUMPTO          00A1
 ENTER           00A1
 ENTCOR          00A1
 PRTREG          00A1
 TERMIN          00A1
 RESUME          00A1
 DMPCOR          00A1
 MASDMP          00A1
 SETBRP          00A1
*L          DRIVERS
 DR1728          0F2D
 CD1729          125B
 TAPEDR          16FA
 FRWA            1822
```

```
  FRWB            18CF
  RECOVT          19A1
  TAPE            1A10
  CARDRD          1A1A
  PRINTR          1B7F
  DISKWD          1D5A
  MASDRV          1F04
  S13001          212C
  SPACE           229F
*M          1713 READER
  DRVMAC          00AF
  S13002          00AF
*M          1713 PUNCH
  DRVMAC          00B2
  S13003          00B2
*S, TIMINT, 7FFF
*S, SNAPE, 7FFF
*S, PARITY, 7FFF
*S, IPROC1, 7FFF
*S, T30, 7FFF
*S, T29, 7FFF
*S, T28, 7FFF
*S, T27, 7FFF
*S, T26, 7FFF
*S, T25, 7FFF
*S, T24, 7FFF
*S, T23, 7FFF
*S, T22, 7FFF
*S, T21, 7FFF
*S, T20, 7FFF
*S, T19, 7FFF
*S, T18, 7FFF
*S, T17, 7FFF
*S, T16, 7FFF
*S, T13, 7FFF
*S, T11, 7FFF
*S, T8, 7FFF
*S, T7, 7FFF
*S, T5, 7FFF
*S, T3, 7FFF
*S, JKIL, 7FFF
*S, RWBA, 7FFF
*S, RW609, 7FFF
*S, DEBUG, 7FFF
*S, DTIMER, 7FFF
*T
```

14. **Set request priorities after installation.**

3.6.5 1721/1722 PAPER TAPE READER DRIVER

Description

The 1721/1722 paper tape reader driver allows data input from the paper tape reader to core memory and interprets eight-level tape only. The reader directly connects to the 1704 computer and is part of the low-speed I/O common synchronizer package.

Installation Requirements

Mass Memory: None.

Core Memory:

driver	216 words
logical unit tables	3 words
diagnostic timer table	1 word
physical equipment table	16 words
	236 words

Procedures

The following installation procedures are unique to the 1721/1722 paper tape reader driver.

1. The equipment code is preset to one for all low-speed I/O common synchronizer devices.

2. The following four-word interrupt entry associated with the low-speed I/O common synchronizer package is assigned to line 1 and must be in LOCORE.

```
LINE1     NUM      0
          RTJ-     ($FE)
          NUM      10
          ADC      EPROC
```

3. Insert in LOG1A:

```
          ADC      PPTRDR
```

4. Insert in LOG1:

```
          ADC      0
```

5. Insert in LOG2:

```
                    ADC           $FFFF
```

6. Enter as externals:

```
                    EXT           PREADI, PTREAD, PTRERR
```

7. Add the PHYSTB to the system tables and parameters using the following coding. The driver priority level is 10; the equipment type is 1; the equipment class is 4:

WORD	LABEL	OP	ADDRESS	COMMENTS
0	PPTRDR	NUM	$120A	
1		ADC	PREADI	
2		ADC	PTREAD	
3		ADC	PTRERR	
4-6		NUM	-1, 0, 0	
7-8		ADC	$A1, $2012	
9-16		BZS	(8)	

8. Add the following entry to the diagnostic timer table (DGNTAB) if time-out surveillance over reader operation is desired:

```
                    ADC           PPTRDR
```

9. Modify MASKT according to instructions in Part III, Section 1.2.3.

3.6.6 1723/1724 PAPER TAPE PUNCH DRIVER

Description

The 1723/1724 paper tape punch driver allows data output from core memory to the paper tape punch. The driver punches eight-level tape only. The punch connects directly to the 1704 computer and is part of the low-speed I/O common synchronizer package.

Installation Requirements

Mass Memory: None.

Core Memory:

driver	207 words
logical unit tables	3 words
diagnostic timer table	1 word
physical equipment table	16 words
	227 words

Installation Procedures

1. The hardware equipment code is preset to one for all low-speed I/O common synchronizer package devices.

2. The following four-word interrupt entry which is associated with the low-speed I/O common synchronizer package should already be assigned to interrupt LINE1 and must contain the following:

```
LINE1     NUM        0
          RTJ-       ($FE)
          NUM        10
          ADC        EPROC
```

3. Into LOG1A enter:

```
          ADC        PPTPCH
```

4. Into LOG1 enter:

```
          ADC        0
```

5. Into LOG2 enter:

```
          ADC        $FFFF
```

6. Enter as an external:

```
          EXT        PUNCHI, PUNCDR, PCHERR
```

7. Insert the following PHYSTB. The driver priority level is 10; the equipment type is 2; and the equipment class is 4.

WORD	LABEL	OP	ADDRESS	COMMENTS
0	PPTPCH	NUM	$120A	
1		ADC	PUNCHI	
2		ADC	PUNCDR	
3		ADC	PCHFRR	
4-6		NUM	-1, 0, 0	
7-8		ADC	$C1, $2024	
9-15		BZS	(7)	

8. If time-out surveillance over punch operation is desired, insert the following entry into the diagnostic timer table:

```
                                        ADC          PPTPCH
```

9. Modify MASKT according to instructions in III.1.2.3.

3.6.7 1729 CARD READER DRIVER

Installation Requirements

Mass Memory: None

Core Memory: Following is the core memory requirement for the 1729:

1729 driver	357 words
system tables	3 words
PHYSTB	17 words
diagnostic timer table	1 word
	378 words of core memory

Installation Procedures

The priority of all low-speed devices (paper tape reader, paper tape punch and teletypewriter) will be changed to a high priority if the 1729 is present, since it is necessary to read the whole card when motion begins. See step 6.

1. Since the 1729 is part of the low-speed package which comes into the computer on interrupt line 1, insert the following:

```
        LINE1       NUM         0
                    RTJ-        ($FE)
                    NUM         $C
                    ADC         EPROC
```

2. Add to LOG1A:

```
                    ADC         CARD29
```

3. Add to LOG1:

```
                    ADC         0
```

4. Add to LOG2:

```
                    NUM         $FFFF
```

5. Insert the following entry point and externals:

```
                    ENT         CARD29
                    EXT         CARDI, CARDR, CDRERR
```

6. The following is a sample PHYSTB for the 1729. Since the 1729 is a high priority device, the low-speed package should be given a priority of 12 when using this driver. Any variation from this example may produce unpredictable results.

WORD	LABEL	OP	ADDRESS	COMMENTS
0	CARD29	NUM	$120C	
1		ADC	CARDI	
2		ADC	CARDR	
3		ADC	CDRERR	
4		NUM	-1	
5-8		NUM	0, 0, $E1, $1872	
9-16		NUM	0, 0, 0, 0, 0, 0, 0, 0	

3.6.8 1728-430 READER-PUNCH DRIVER

Description

The 1728-430 reader-punch driver executes at high priority while data is read in and at low priority while data is interpreted, converted, and packed. This minimizes possible destructive interaction with other concurrently executing drivers.

Installation Requirements

Mass Memory: None

Core Memory: Following is the core memory requirement:

driver	861
PHYSTB and buffer	128
system tables	4
	993 words of core memory

Installation Procedures

1. Use the Macro Assembler to assemble the 1728-430 driver routine and to produce a relocatable binary tape.

2. Following is an example of an interrupt trap which must be inserted with x as the interrupt line on which the 1728-430 is to be connected:

```
          EXT    I1728
LINEx     NUM    0
          RTJ-   ($FE)
          NUM    13
          ADC    I1728
```

3. Declare the following names as external and entry symbols in SYSBUF anywhere before END and after NAM:

```
          EXT    IN1728
          EXT    CN1728
          EXT    EX1728
          EXT    FF1728
          EXT    CM1728
          ENT    I1728
```

4. Insert into LOG1A the label associated with word 0 of PHYSTB. x is the interrupt line to be connected to the 1728-430.

```
After:    EQU    Lx(*)
Insert:   ADC    label
```

5. An interrupt response routine is advisable for the card reader to save time. The following will suffice:

```
          I1728    LDQ        =X label
                   JMP*       (label + 2     JMP  TO  CONTINUATOR
```

6. Add a zero cell to LOG1 at the logical unit position corresponding to the 1728-430 entry made in LOG1A using the following form:

```
                   ADC        0
```

7. Add to LOG2 the following code at the logical unit position which corresponds to the 1728-430 entry made in LOG1A:

```
                   NUM        $FFFF
```

8. Modify MASKT according to instructions in part III, section 1.2.3.

9. Insert the following PHYSTB consisting of 43 words after the last PHYSTB inserted in the system. After the word 43, insert the 80-word buffer. Place a label on word 0 to match the LOG1A entry.

WORD	LABEL	OP	ADDRESS	SIGNIFICANCE	
0	label	NUM	$12xx	xx is the initiator priority level which should equal the priority level of the interrupt line in program LOCORE	
1		ADC	IN1728	initiator entry	
2		ADC	CN1728	continuator entry	
3		ADC	EX1728	hangup entry	
4		NUM	-1	diagnostic clock setting	
5		NUM	0		
6		NUM	0		
7		NUM	$0421	bits 15-11	00000
				bits 10-7	Q which is the 1728-430 equipment number
8		NUM	$08C6	magnetic tape equipment type to allow motion control requests	
9-13		NUM	0, 0, 0, 0, 0		
14		NUM	$wxyz	w is 8 if error is to be signaled when switch is attempted from read to punch or vice-versa xyz represents the 12 bits of column one which is to be interpreted as an end of file card	

WORD	LABEL	OP	ADDRESS	SIGNIFICANCE
14 (continued)				Example: $0006 would mean $\overset{7}{8}$ PUNCH is end of file and no read/punch switch checking wanted. $8006 would mean $\overset{7}{8}$ PUNCH is end of file and read/punch switch checking is wanted.
15		ADC	BF1728	address of an 80-word BZS in SYSBUF
16-24		NUM	0, 0, 0, 0, 0, 0, 0, 0, 0	
25	NF1728	RTJ-	($F4)	
26		NUM	$Cxx	x = initiator priority level
27		ADC	CCC	
28	TH1728	ADC	0	
29		NUM	$18FB	
30		NUM	5	
31		ADC	MS1728	
32		JMP	FF1728	
33	CCC	LDA*	IIIIII	
34		STA-	I	
35		JMP	CM1728	
36	IIIIII	ADC	label	
37	MS1728	NUM	$5351	
38		NUM	$2020	
39	AA1728	NUM	0	
40		NUM	$2C20	
41	BB1728	NUM	0	
42		BZS	BF1728(80)	

3.6.9 1729-2 CARD READER DRIVER

Description

The 1729-2 card reader driver executes at high priority while data is read in and executes at low priority while data is interpreted, converted, and packed. This minimizes possible destructive interaction with other concurrently operating drivers.

Installation Requirements

Mass Memory: There is no mass memory requirement.

Core Memory:

driver	454
PHYSTB and buffer	108
system tables	4
	566 words of core memory

Installation Procedures

1. With x as the interrupt line on which the 1729-2 is to be connected, insert the following in the interrupt trap area of LOCORE:

```
         EXT     I1729
LINEx    NUM     0
         RTJ-    ($FE)
         NUM     $D
         ADC     I1729
```

2. Insert the following names as external symbols in SYSBUF anywhere between NAM data and END:

```
         EXT     EX1729
         EXT     IN1729
         EXT     CN1729
```

3. Enter into LOG1A the label associated with word 0 of the PHYSTB:

```
         ADC     label
```

4. Insert in LOG1 a zero cell at the index position corresponding to the 1729-2 entry made in LOG1A.

```
         ADC 0
```

5. Add to the LOG2 at the logical unit position corresponding to the 1729-2 entry made in LOG1A:

```
         NUM     $FFFF
```

6. Modify the MASKT according to the instructions in III.1.2.3.

7. Insert this 25 word PHYSTB with a label on word 0:

WORD	LABEL	OP	ADDRESS	SIGNIFICANCE
0	label	NUM	$120x	x is the initiator priority level which should equal the priority level of the interrupt line in program LOCORE.
1		ADC	IN1729	initiator entry
2		ADC	CN1729	continuator entry
3		ADC	EX1729	hang-up entry
4		NUM	-1	diagnostic clock setting
5		NUM	0	logical unit
6		NUM	0	call parameter list
7		NUM	00000 | Q | 0100001 15 10 7 6 0	Q is the 1729-2 equipment number
8		NUM	$19D2	equipment type
9-13		NUM	0, 0, 0, 0, 0	
14		NUM	$xxxx	xxxx represents the 12 bits of column one which are to be interpreted as an end of file card. For example, $0006 would mean that $\substack{7\\8}$ PUNCH is an end of file.
15		ADC	BF1729	BF1729 is the address of an 80-word BZS in SYSBUF program.
16-24		NUM	0, 0, 0, 0, 0, 0, 0, 0, 0	
25		BZS	BF1729(80)	80-word buffer

8. Enter the following interrupt response routine into SYSBUF:

```
                ENT       I1729
        I1729   LDQ       =CD1729
                JMP*      (CD1729 + 2)
```

3.6.10 1731/1732-601/608/609 MAGNETIC TAPE DRIVERS

1731/1732 Buffered Magnetic Tape Driver

Installation Requirements: The following modules, with corresponding memory requirements, are necessary:

Modules	Memory
TAPDRB	466 words
FRWAB	236 words
FRWBB	251 words
RECVTB	123 words
TAPE	10 words
	1086 words

The following is an optional module:

RW609B	109 words

Installation Procedures: Make the standard changes to LOCORE and SYSBUF which are listed in the introduction to 3.6. Also add space to AREAC of the space driver for a buffer area which is three times the maximum buffer size. The following procedures outline only those items which are unique to the 1731/1732 Buffered Magnetic Tape Driver.

1. Insert the following in the interrupt trap area of LOCORE using the desired interrupt line:

```
LINEx   NUM     0
        RTJ-    ($FE)
        NUM     11
        ADC     INT601
```

2. Insert into LOG1A the first word label in the position corresponding to the interrupt line to be used:

```
        EQU     Lx(*)
        ADC     TPPDR1
        ADC     TPPDR2
```

3. Insert in LOG1:

```
        ADC     0
        ADC     0
```

4. Insert in LOG2:

```
        NUM     $FFFF
        NUM     $FFFF
```

5. Insert a PHYSTB similar to the one below:

WORD	LABEL	OP	ADDRESS	COMMENTS
0	TPPDR1	NUM	$12BB	PHYSTB FOR MAG TAPE DRIVER
1		ADC	TAPDRB	INITIATOR
2		ADC	TAPCB	CONTINUATOR
3		ADC	TAPHB	ERROR
4-6		NUM	-1, 0, 0	
7		NUM	$1381	EQUIPMENT 7
8-13		NUM	$896, 0, 0, 0, 0, $414	
14		ADC	TPPDR2	
15		ADC	0	
16		NUM	96	MAXVAL MAXIMUM RECORD SIZE

The following words are unique to the 1731/1732 buffered magnetic tape drivers:

Word	Bit	Description
13		information to select specific tape unit and initially set the recording mode and density
	15-11	not used
	10	set to 0 initially
	9-7	physical unit number (0-7)
	6	not used
	5-3	tape density
	5	if set to 1, 800 bpi
	4	if set to 1, 556 bpi
	3	if set to 1, 200 bpi
	2-1	parity (recording mode)
	2	if set to 1, binary (odd parity)
	1	if set to 1, BCD (even parity)
14		Thread. For the tape driver, the thread links all magnetic tape entries which are connected to a single controller. For example: if three tape units are connected to the controller and the labels for their PHYSTB entries are TAG1, TAG2, and TAG3, the thread in entry TAG1 would contain ADC TAG2, the thread in TAG2 would contain ADC TAG3, and the thread in entry TAG3 would contain ADC TAG1.

15	15-0	set to 0
16		MAXVAL is the maximum number of words per record

2. Since EPROC cannot be used as an interrupt response routine for the buffered magnetic tapes, a user supplied routine must check interrupt conditions and terminate the buffer on a short read (1706 is busy). Following is a sample routine for the buffered 601.

```
LABEL       OP      ADDRESS       COMMENTS

INT601      LDQ*    PHYSTB        SET PHYSTB ADDRESS
            LDQ-    7,Q
            LDA-    NZERO+11
            LAQ     Q
            ADQ-    ONEBIT+11
            NOP     0
            INP     -1            TERMINATE THE BUFFER
            LDQ*    PHYSTB
            LDA-    2,Q           SET CONTINUATOR ADDRESS
            STA-    I
            JMP-    (I)
PHYSTB      ADC     TPPDR1
```

1731-601 Unbuffered Magnetic Tape Driver

Installation Requirements: The following modules, with corresponding memory, are necessary:

Modules	Memory
TAPEDR	310 words
FRWA	186 words
FRWB	210 words
RECOVT	114 words
TAPE	10 words
	830 words

The following is an optional module:

RWBA	106 words

```
NUM      11
ADC      EPROC
```

2. Insert into LOG1A the first word label in the position corresponding to the interrupt line used by the 601. For example, after EQU Lx(*) insert:

```
ADC      TPPDR1
ADC      TPPDR2
```

3. Insert into LOG1:

```
ADC      0
```

4. Insert into LOG2:

```
NUM      $FFFF
```

5. Insert a PHYSTB for the 601 similar to the following example:

```
WORD     LABEL     OP       ADDRESS            COMMENTS
                   EXT      TAPEDR, TAPEC,
                            TAPEH
0        TPPDR1    NUM      $120B
1                  ADC      TAPEDR
2                  ADC      TAPEC
3                  ADC      TAPEH
4-6                NUM      -1, 0, 0
7                  NUM      $381               EQUIPMENT 7
8                  NUM      $896
9-13               NUM      0, 0, 0, 0, $414
14                 ADC      TPPDR2
15                 NUM      0
```

1732-608 Unbuffered Magnetic Tape Driver

Installation Requirements: See the requirements listed for the 1731-601.

Installation Procedures: Make the standard changes listed in 3.6 along with the following procedures unique to the 1732-608:

1. Insert the following in the interrupt trap area of LOCORE using the desired interrupt line:

```
LINEx           NUM         0
                RTJ-        ($FE)
                NUM         11
                ADC         INT608
```

2. Insert into LOG1A the first word label in the position corresponding to the interrupt line used by the 608. For example, after EQU Lx(*) insert:

```
                ADC         TPPDR1

                ADC         TPPDR2
```

3. Insert into LOG1:

```
                ADC         0
```

4. Insert into LOG2:

```
                NUM         $FFFF
```

5. Insert an interrupt processor similar to the following before the 608 PHYSTB:

```
LABEL           OP          ADDRESS         COMMENTS
                ENT         INT608
                EQU         TPEDEV(1)       NUMBER OF TAPE DEVICES ON
                                            CONTROLLER-1
INT608          ENA         TPEDEV
REPEAT          STA-        I
                LDQ*        TABLE,I
                LDQ-        7,Q
                INP         NOTIT-*
                ALS         13
                SAM         OK608-*-1
NOTIT           NOP         0
                LDA-        I
```

1732-608 Unbuffered Magnetic Tape Driver

Installation Requirements: See the requirements listed for the 1731-601.

Installation Procedures: Make the standard changes listed in 3.6 along with the following procedures unique to the 1732-608:

1. Insert the following in the interrupt trap area of LOCORE using the desired interrupt line:

```
LINEx          NUM       0
               RTJ-      ($FE)
               NUM       11
               ADC       EPROC
```

2. Insert into LOG1A the first word label in the position corresponding to the interrupt line used by the 608. For example, after EQU Lx(*) insert:

```
               ADC       TPPDR1

               ADC       TPPDR2
```

3. Insert into LOG1:

```
               ADC       0
```

4. Insert into LOG2:

```
               NUM       $FFFF
```

5. Enter a PHYSTB similar to the following:

WORD	LABEL	OP	ADDRESS	COMMENTS
		ENT	TPPDR3	
		EXT	TAPEDR, TAPEC, TAPEH	
0	TPPDR3	NUM	$120B	
1-3		ADC	TAPEDR, TAPEC, TAPEH	
4-6		NUM	-1, 0, 0	
7		NUM	$281	EQUIPMENT 5
8-12		NUM	$A46, 0, 0, 0, 0	

```
13                NUM          $514
14                ADC          TPPDR4
15                ADC          0
```

6. Word 14 in the PHYSTB is a thread word which links the PHYSTBs of all magnetic tape drives on a controller. If only one equipment table is present, it contains a thread to itself as:

```
14                ADC          TPPDR1
```

1732-608/609 Magnetic Tape Driver

Description: The 1732-608/609 driver can execute in a formatted or unformatted, buffered or unbuffered mode. To generate the release relocatable binary tape, use the following macro call:

```
TPDRGN          CORE,UNBUF,608,FORM,ERR,192,11
```

Installation Requirements: Specified below is the length of the driver as it is assembled from the COSY source tape; the length can vary according to the assembly options chosen.

Driver	348
System tables and parameters	4
PHYSTB	17
	369 words of memory

Installation Procedures

1. For an unbuffered driver, insert an entry similar to the following into the appropriate interrupt trap area of LOCORE.

LABEL	OP	ADDRESS	COMMENTS
LINEx	NUM	0	
	RTJ-	($FE)	
	NUM	11	PRIORITY LEVEL
	ADC	EPROC	

2. For a buffered driver, an interrupt response routine must be inserted instead of using EPROC. Following is an example of an interrupt response routine for a buffered driver. This routine can be placed in SYSBUF.

```
 LABEL   OP      ADDRESS                  COMMENTS

 INT608  LDQ*    PHYSTB                   SET PHYSTB ADDRESS
         LDQ-    7,Q
         LDA-    NZERO+11
         LAQ     Q
         ADQ-    ONEBIT+11
         NOP     O
         INP     -1                       TERMINATE THE BUFFER
         LDQ*    PHYSTB
         LDA-    2,Q                      SET CONTINUATOR ADDRESS
         STA-    I
         JMP-    (I)
PHYSTB   ADC     TPPDR1
```

An interrupt trap entry for the buffered 1732-608/609 using the interrupt response routine given above is:

```
LINEx  NUM     0
       RTJ-    ($FE)
       NUM     11
       ADC     INT608               INTERRUPT RESPONSE ROUTINE FOR
                                    BUFFERED 608
       EXT     INT608
```

3. Insert into the LOG1A table after EQU Lx(*):

```
       ADC     TAPDR1
```

4. Insert in LOG1:

```
       ADC     0
```

5. Insert in LOG2:

LABEL	OP	ADDRESS	COMMENTS
	NUM	$FFFF	

6. Insert in the diagnostic timer table:

```
          ADC     TAPDR1
```

7. Insert a PHYSTB similar to the following:

WORD	LABEL	OP	ADDRESS	SIGNIFICANCE
		EXT	TAPINT, TAPCON, TPHANG	
0	TAPDR1	NUM	$120B	Driver at level 11
1		ADC	TAPINT	Initiator
2		ADC	TAPCON	Continuator
3		ADC	TPHANG	I/O hang up
4		NUM	-1	Diagnostic clock
5		NUM	0	
6		NUM	0	
7		NUM	xxxx	$1281 for buffered driver $281 for unbuffered driver
8		NUM	xxxx	$A66 for 608 $A76 for 609
9		NUM	0	Status word 1
10		NUM	0	
11		NUM	0	
12		NUM	0	
13		NUM	xxxx	$4C0 for select unit 1 $440 for select unit 0

Bits	
15-11	Unused; set to zero
10	Select; set to one
9-7	Tape unit number (0-7)
6	Assembly mode of data transfer; set to one
5	Select 200 BPI; set to zero
4	Select 556 BPI; set to zero

WORD	LABEL	OP	ADDRESS	SIGNIFICANCE
				3 Select 800 BPI; set to zero
				2 Binary (odd parity); set to zero
				1 BCD (even parity) 608 only; set to zero
				0 Not used
14		ADC	TAPDR2	Address of next PHYSTB
15, 16		NUM	0,0	Temporary storage

8. DR1732 is coded as a macro skeleton. To parameterize the driver for a particular configuration, prepare a COSY control deck as follows:

LABEL	OP	ADDRESS	COMMENTS
DR1732	DCK/	I=lu, H=lu	
	DEL/		
TPDRGN	$P_1, P_2, P_3, P_4, P_5, P_6, P_7$		

The formal parameters P_1 through P_7 are defined as follows:

P_1 Defines the residency of the driver

CORE For core resident

MASS For mass memory resident. When MASS is specified the following additional deck card is needed.

TAPCOR DCK/ I=lu, H=lu

P_2 Defines the method of data transfer:

BUF For buffered transfers using a 1706 buffered data channel

UNBUF For unbuffered transfers using the A Q channels

P_3 Defines the type of tape drives which will be in the system: 608, 609, or BOTH

P_4 Defines the type of read/write requests to be processed

FORM Only formatted requests are to be processed

REG Only regular requests are to be processed

BOTH Formatted and regular requests are to be processed

P_5 Defines whether error recovery for parity errors will be attempted

ERR Recovery will be attempted

NOERR The error bits are set; the request is completed

P_6 Defines the maximum tape record size for 608 units. If blank, 96 words are assumed. The standard binaries were made using 192. This was done to allow COSY to run records which are two sectors in length or 192 words.

P_7 Defines the priority level at which the driver is to operate. This level should be in the range of 5 to 14 and should also be used in the interrupt trap area and the PHYSTB.

9. To obtain source, decosy the necessary decks specified above. See 1700 COSY/MSOS Reference Manual 60237100.

10. Assemble the source of DR1732. Also assemble TAPCOR if the driver is to be mass resident.

11. If the driver is to be in core resident, insert it in the source tape before the SPACE module as follows:

```
*L
   DR1732        (binaries)
```

12. If the driver is to be on mass resident, modify the SPACE module. After assembling DR1732 for the relevant site configuration, note the number of core locations which the driver requires. The value specified by N4 is increased by the length of the driver (length of DR1732 is xxxx) as follows:

```
LABEL    OP      ADDRESS                  COMMENTS

SPACE    DCK/     I=lu, H=lu

         DEL/

         EQU     N4($0+xxxx)
```

13. When using the mass memory version, replace the SPACE module, insert TAPCOR as a core resident program and DR1732 as mass memory with the appropriate ordinal assignments. The following core structure indicates where to place these items.

```
*Y.... (if any)
*YM...., TAPMAS,nn

*L
 LOCORE
 SYSBUF
•
•
•
•
*L
 TAPCOR (core resident 1732 driver module)
          Must be placed before the mass memory portion.
•
•
•
•
*M  system ordinals
•
•
•
•
*M  (nnth *M statement)
```

```
DR1732
•
•
•
*T
•
•
•
*S, nn, 11, M
•
•
•
*U
```

14. After updating the installation tape, install the system as it would normally be installed.

Special System Modification:

Under certain tape motion control requests it is possible to receive a JO2. To prevent this error, make the following modifications to the protect processor.

```
LABEL     OP      ADDRESS              SIGNIFICANCE

PROTEC    DCK/    I=lu,H=lu
          INS/    29
          EQU     LPMSK(2)
          DEL/    294
X14       LDA     TIME,Q
          DEL/    320
Y         RTJ     G                    Stock and check C
          DEL/    379,380
          SAZ     NEWTAP               Check for D606 *2
          JMP     P1
NEWTAP    LDA-    8,Q
          AND     =N$7F0               Equip type
          ARS     4
          INA     -39                  Code for 609
          SAZ     LA609
          INA     1
          SAN     W31                  Code for 608
          JMP*    W7
```

```
LABEL     OP      ADDRESS           SIGNIFICANCE

LA609     LDA-    5,I               Pick up P1, P2, P3, density
          SAM     LA609A
          AND-    NZERO+4           Clear density bits
          STA-    5,I
LA609A    JMP*    W5
          DEL/    398
W7        LDA-    5,I               Pick up P1, P2, P3, density
          SAM     W7A
          AND-    LPMSK+4           Check for illegal density
          INA     -4
          SAP     W7B
W7A       JMP*    W5
W7B       JMP*    Z0
          END/
```

3.6.11 1738-853/854 DISK DRIVER (DISKWD)

Description

The 1738-853/854 disk driver provides the capability for data transfer to and from the disk as a mass memory device. Additionally, the disk driver handles the transfer of mass-memory-resident programs into core as a result of SCHDLE requests. This driver permits word addressability simulation.

The 1738 disk interface uses the direct storage access bus for its input/output to provide completely buffered operation. The disk driver complements this capability by requiring control upon end-of-operation or error condition only as indicated by an interrupt.

Installation Requirements

Mass Memory: None

Core Memory: Core requirements for one disk driver:

driver	425 words
system tables and parameters	
logical unit tables	3 words
physical equipment table	22 words
interrupt response routine	3 words
OVRLAY subroutine	8 words
interrupt trap region	4 words
	465 words

Core requirements for a two-disk driver are the same as for a one-disk driver except for the following deviations. Logical unit tables require a total of 6 words and the two PHYSTB's require a total of 44 words.

Installation Procedures

The following instructions apply to the installation of one disk storage driver. (When two drives are used, the second drive must have a unit address of 1, a separate PHYSTB must be assigned, and entries in the diagnostic timer table and in LOG1A, LOG1, LOG2 must be assigned.)

1. Insert the following four-word interrupt entry using the desired interrupt line in place of x.

```
LABEL   OP      ADDRESS
LINEx   NUM     0
        RTJ-    ($FE)
        NUM     9
        ADC     EPROC
```

2. Insert in LOG1A an entry for each disk.

Using one disk

LABEL	OP	ADDRESS
	ADC	DISK0

Using two disks

```
        ADC     DISK0
        ADC     DISK1
```

3. Insert in LOG1 an entry for each disk.

   Using one disk

```
        ADC     0
```

Using two disks

```
        ADC     0
        ADC     0
```

4. Insert in LOG2 an entry for each disk.

   Using one disk

```
        NUM     $FFFF
```

Using two disks

```
        NUM     $FFFF
        NUM     $FFFF
```

5. Insert the following coding in SYSBUF:

```
        ENT     DISK0
        EXT     DKINTR, DKCONT, DKDIAR
```

6. Insert the following PHYSTB when installing one disk driver:

WORD	LABEL	OP	ADDRESS	COMMENTS
0	DISK0	NUM	$1209	
1		ADC	DKINTR	
2		ADC	DKCONT	
3		ADC	DKDIAR	
4-6		NUM	-1, 0, 0	
7		ADC	$181	

WORD	LABEL	OP	ADDRESS	SIGNIFICANCE
8		ADC	$xxxx	$1086 if using the 853 $1096 if using the 854
9		ADC	$200	
10-15		BZS	(6)	
16		ADC	$11A	
17		ADC	0	
18-21		BZS	(4)	18 to 21 overlay calls moved to here

Insert the following PHYSTB for two disks per controller.

0	DISK0	NUM	$1209	Library and scratch disk
1		ADC	DKINTR	
2		ADC	DKCONT	
3		ADC	DKDIAR	
4-16		NUM	-1, 0, 0, $181, $1056, $200, 0, 0, 0, 0, 0, 0, $11A	
17		ADC	DISK1	
18-21		BZS	(4)	Words 18 to 21 are for overlay calls moved here
	*			
	*			
0	DISK1	NUM	$1209	
1		ADC	DKINTR	
2		ADC	DKCONT	
3		ADC	DKDIAR	
4-16		NUM	-1, 0, 0, $181, $1056, $200, 0, 0, 0, $1000, 0, 0, $31A	
17		ADC	DISK0	
18-21		BZS	(4)	Words 18-21 overlay calls moved here

7. If the time-out surveillance is desired, insert the following entries into the diagnostic timer table.

For one disk:

```
ADC     DISK0
```

For two disks:

```
ADC     DISK0
ADC     DISK1
```

8. DKDIAG, the disk diagnostic subroutine, resides in SYSBUF and handles all error recovery for the driver. If the user wishes to supply his own routine, that routine must comply with the following requirements which apply to the present standard version.

   If a read of a system directory program into allocated core produces an error, this allocated core must be released.

   It can take diagnostic action as desired.

   It is a closed subroutine which is entered by a RTJ instruction.

   It must be entered with the disk PHYSTB address in the I register to allow access to the request parameters for diagnostic action.

   The standard version types the message:

   MASS MEM ERR code

0	Time-out error; 1738 malfunction; no completion interrupt occurred as a result of disk operation initiation.
1	Internal or external reject occurred on an INP or OUT instruction. Possible causes are: equipment turned off, erroneous equipment code, or 1738 malfunction.
2	Alarm.
3	Parity error.
4	Checksum error

The nature of the error is also indicated in the Q register upon entry to DKDIAG with one of the codes listed above. If an error occurs on a SCHDLE request, the assigned core area is released; no completion address is scheduled. When DKDIAG is finished, it returns control to the driver with the I register intact.

3.6.12 1742 LINE PRINTER DRIVER

Installation Requirements

1742 driver	478
System tables	3
PHYSTB	17
Diagnostic timer table	1
	499 words

Installation Procedures

1. Insert the following into the interrupt trap area, replacing x with the interrupt line to which the 1742 is to be connected.

LABEL	OP	ADDRESS
LINEx	NUM	0
	RTJ-	($FE)
	NUM	10
	ADC	EPROC

2. Insert in LOG1A after EQU LX(*):

```
                    ADC     LP1742
```

3. Insert in LOG1:

```
                    ADC     0
```

4. Insert in LOG2:

```
                    NUM     $FFFF
```

5. Add the following entries:

```
                    ENT     LP1742
                    EXT     PRINTI, PRINTC, PRIERR
```

6. A sample PHYSTB for the 1742 is:

WORD	LABEL	OP	ADDRESS
0	LP1742	NUM	$120A
1		ADC	PRINTI
2		ADC	PRINTC
3		ADC	PRIERR
4		NUM	-1
5-6		NUM	0, 0
7		NUM	$781
8		NUM	$28B4
9-16		NUM	0, 0, 0, 0, 0, 0, 0, 0

To utilize the line printer for FORTRAN format (FORTRAN line printer), make the following additions to SYSBUF:

1. To LOG1A, add the following statement at the appropriate logical unit:

```
          FLP       ADC     LP1742
```

2. Add to the LOG1 for the appropriate logical unit:

```
                    ADC     $4000               LU 1742 FTN LINE PTR
```

Modify the entry for the standard line printer to:

```
ADC      $4000
```

3. Add to LOG2 for the logical unit corresponding to the LOG1A and LOG1 tables:

```
NUM      $FFFF               LU 1742 FTN LINE PTR
```

4. Add to the 1742 PHYSTB:

WORD	LABEL	OP	ADDRESS	COMMENTS
17		ADC	FLP-LOG1A	1742 FTN LINE PTR

3.6.13 1745-2 DISPLAY DRIVER 1.0

Installation Requirements

Mass Memory: No mass memory is necessary.

Core Memory:

Buffered driver	676	Unbuffered driver	584
PHYSTB and interrupt response routines	14	PHYSTB and interrupt response routines	97
System tables	4	System tables	4
	794		685

Installation Procedures

1. Since the display is capable of interrupting at any time (if someone pushes the SEND key), it is recommended that the display driver execute at a priority level equal to or lower than any other device on the 1706 so as not to interrupt activity between the 1706 and other peripheral gear which may be attached to the 1706. Also, the 1706 and the 1745-2 should have equal priority. Write and install in the SYSBUF part of the operating system a routine similar to the one below (if one is not already available) to handle interrupts from the 1706 and to direct these interrupts to the display driver.

LABEL	OP	ADDRESS	COMMENTS
	ENT	CRT45	
	ENT	CRTINT	
CRTINT	LDQ	=N$2000	CHECK STATUS ON 1706 (USE WITH BUFFERED DRIVER ONLY)
	INP	-2	(USE WITH BUFFERED DRIVER ONLY)
	ALS	13	CHECK INTERRUPT BIT (USE WITH BUFFERED DRIVER ONLY)

```
LABEL    OP      ADDRESS           COMMENTS

         SAM     CRTGO-*-1         (USE WITH BUFFERED DRIVER ONLY)
CRT45    LDQ     =XCRT451
         LDQ-    7,Q               STATUS ON CRT DEVICE
         INP     -1
         ALS     13                CHECK INTERRUPT BIT
         SAP     3
         ALS     5                 LOOK FOR SEND REQUEST
         SAP     CRTGO-*-1
         JMP*    CKLUN
         LDA     =N$AAAA           INDICATES HARDWARE ERROR
         JMP*    *-2               FALSE INTERRUPT
CRTGO    LDA     =XCRT451
         STA-    I
         LDA-    5,I               LOOK FOR LUN
         SAZ     NOTFND-*-1        LOOK FOR ANOTHER LUN
         LDA-    9,I               CK WORD 9 OF PHYSTB
         ALS     14                CHECK FORMAT BIT
         SAP     FOUND-*-1         FOUND TIME INTERRUPTING LUN
         ALS     1                 CK READ OR WRITE
         SAM     FOUND-*-1
         ALS     10                CK BIT6, CRT WAIT FOR SEND
         SAM     FOUND-*-1
NOTFND   LDA-    14,I              PICK UP THREAD
         SUB     =XCRT451
         SAZ     CRTGO2-*-1
         LDA-    14,I
         JMP*    CRTGO+2
FOUND    LDQ-    I
         LDA-    2,I
         STA-    I
         JMP-    (I)               JMP TO CONTINUATOR
```

```
LABEL     OP        ADDRESS              COMMENTS
CRTGO2    LDA       =N$6666              ERROR, UNABLE TO FIND
          JMP*      *-2                  LUN RESPONSIBLE FOR INTERRUPT
CKLUN     LDQ       =XCRT451
          LDQ-      7,Q
          INQ       1                    CHECK FOR STATION NUMBER
          NOP       0
          INP       -1
          AND       =N$FFF0              DROP STATUS SWITCH BITS IF ANY
          STA*      SAVNUM
          LDA       =XCRT451
CKSTA     STA-      I
          LDA-      13,I                 LOOK FOR STATION NUMBER
          SUB*      SAVNUM               COMPARE STATION NUMBERS
          SAN       1
          JMP*      FOUND
          LDA-      14,I
          SUB       =XCRT451
          SAN       1
          JMP*      CRTGO2
          LDA-      14,I
          JMP*      CKSTA
SAVNUM    NUM       0
```

Add the following entries when using a buffered driver:

```
          ENT       KILL06
KILL06    STQ*      KEEPQ
          STA*      KEEPA
          LDQ       =N$1800              TERM BUFFER ON 1706 NO. 1
          NOP       0                    AT TIME OF 1745-2 INTERRUPT
          INP       -1
          LDA*      KEEPA
          LDQ*      KEEPQ
```

```
LABEL     OP        ADDRESS                  COMMENTS
          JMP*      CRT45
KEEPA     NUM       0
KEEPQ     NUM       0
```

2. For a buffered driver: since both the 1706 and the 1745-2 End-of-Operation interrupts are enabled during a read operation to terminate the read at End-of-Buffer or End-of-Message (whichever comes first), include a few words of code in SYSBUF (see sample KILL06 routine) to suppress the 1706 if an End-of-Message interrupt occurs from the display controller. Sample coding in LOCORE interrupt trap area to accomplish this is:

```
LABEL     OP        ADDRESS                  COMMENTS
          EXT       CRTINT
LINE2     NUM       0                        INTERRUPT LINE2 ENTRY FOR 1706
          RTJ-      ($FE)                    GO TO COMMON INT. HANDLER
          NUM       9                        PRIORITY LEVEL
          ADC       CRTINT                   1706 BUFF DATA CHAN
          EXT       KILL06
LINE13    NUM       0                        INTERRUPT LINE 13 ENTRY FOR 1745-2
          RTJ-      ($FE)                    GO TO COMMON INT. HANDLER
          NUM       9                        PRIORITY LEVEL
          ADC       KILL06                   TERMINATE 1706
```

For an unbuffered driver, insert coding similar to that below in the Interrupt Trap Area of LOCORE:

```
          EXT       CRT45
LINE13    NUM       0                        INTERRUPT LINE 13
          RTJ-      ($FE)                    GO TO COMMON INT. HAND.
          NUM       9                        PRIORITY LEVEL
          ADC       CRT45                    USER INTERRUPT PROCESSOR
```

3. Insert coding similar to that below in LOG1A. To make the CRT a system comment device, remove the location symbol SCD from the teletype reference in LOG1A and place it in front of the desired CRT reference (see the second and third lines in the example below). If the CRT is used as the comment device, it should execute at the same priority level as the standard comment device does.

```
LABEL     OP        ADDRESS                  COMMENT
          EQU       L13(*)                   INTERRUPT LINE13
          ADC       CRT451                   NAME OF DISPLAY PHYSTB ENTRY
          ADC       CRT452                   (UP TO 12 DISPLAY STATIONS PER LINE)
          etc.
```

4. Into LOG1 insert a card to signify the use of an alternate unit. The following sample entry assumes that a 1745-2 display is logical unit 10 and that the standard comment device is the alternate (LU4). At the location corresponding to logical unit 10, add:

NUM $4004

```
|0|1| 0 0 0 0 | 0 0 0 0 0 0 0 1 0 0|

15 14           9                  0
```

Bits	Significance
14	If 1, indicates a shared device
9-0	Alternate logical unit number; the CRT alternate is the teletypewriter

5. Insert coding similar to the following in LOG2:

LABEL	OP	ADDRESS	COMMENT
	NUM	$FFFF	

6. Modify the MASKT to indicate the priority level and the interrupt line for the 1745-2 Display Controller as described in III.1.2.3.

7. Insert in the DGNTAB:

ADC CRT451

8. Declare the following external:

EXT DDINIT, DDCONT, DDDIAG

9. One entry in the PHYSTB is necessary for each display station. Following is a sample coding for a PHYSTB entry:

WORD	LABEL	OP	ADDRESS	COMMENT
0	CRT451	NUM	$1299	
1		ADC	DDINIT	
2		ADC	DDCONT	
3		ADC	DDDIAG	
4-6		NUM	-1, 0, 0	
7		NUM	xxxx	$1x01 FOR A BUFFERED DRIVER $0x01 FOR AN UNBUFFERED DRIVER
8		NUM	$E6	
9-12		NUM	0, 0, 0, 0	
13		NUM	$0040	PHYSICAL STATION NUMBER, BITS 9-6

WORD	LABEL	OP	ADDRESS	COMMENTS
14		ADC	CRT452	THREAD OF NEXT DEVICE
15		NUM	11	STATION LOGICAL UNIT NUMBER IN BITS 3-0
16-17		NUM	0,0	TEMPORARY STORAGE USED BY DRIVER
18		NUM	$2000	LOAD INTO THE Q REGISTER TO OBTAIN THE 1706 STATUS. THIS WORD IS FOR THE BUFFERED DRIVER ONLY

10. RD1745 is a routine which handles SEND interrupts when there is no current request for the display driver. Install the RD1745 routine in the operating system along with the 1745-2 display driver. To ignore the SEND request, either:

    Immediately exit to the dispatcher, or

    Write an error message and then exit, or

    Schedule one of several process programs (the choice of which program to schedule depends on the status switch settings at interrupt time).

    In the following sample RD1745 routine:

    If status switch 3 is set, a mass memory resident routine is scheduled.

    If only status switch 1 is set, the driver interprets the condition as a manual interrupt request.

    If any other combination of status switches is set, an error message appears on the display and an exit is made to the dispatcher.

LABEL	OP	ADDRESS	COMMENTS
	NAM	RD1745	
	ENT	RD1745	
	EQU	FWRITE($C00), RP($10), CP(1)	
	EQU	AMONI($F4), ADISP($EA)	
	EQU	N000F(6)	
	EXT	USERXY	
STORQ	NUM	0	
STORQL	NUM	0	
RD1745	NOP	0	
	STQ*	STORG	SAVE LUN + STATUS
	LLS	8	
	QRS	8	

```
LABEL   OP      ADDRESS                  COMMENTS
        STQ*    STORQL                   SAVE LUN ONLY
        AND-    N000F                    MASK OFF STATUS BITS IN A
        INA     -4                       CK FOR SWITCH 3 ONLY SET(BIT 2)
        SAN     A-*-1                    IF OTHER SWITCHES, WRITE ERROR
                                         MES
        LDQ*    STORQ
        RTJ-    (AMONI)                  SCHEDULE USER PROCESS TYPE PRG

        NUM     $1255                    WHICH RESIDES ON MASS MEMORY
        ADC     (USERXY)                 NAME OF USER PROGRAM TO BE CALLED
                                         FROM DISK
        JMP-    (ADISP)                  EXIT TO DISPATCHER
A       RTJ-    (AMONI)                  WRITE ERROR MESSAGE TO COMMENT
                                         DEVICE
        ADC     FWRITE+RP*5+CP*5
        NUM     0                        COMPLETION ADDRESS
        NUM     0                        THREAD
        NUM     $8FC                     COMMENT DEVICE
        NUM     17                       LENGTH OF MESSAGE
        ADC     MES                      FWA OF MESSAGE
        JMP-    (ADISP)                  EXIT TO DISPATCHER
MES     ALF     17, INCORRECT STATUS SWITCH SELECTION.
        END
```

11. Optional coding for space module: since a SEND interrupt is capable of being generated at any time (even when the computer is not running), and since this interrupt remains active until it is either processed by the computer or manually master cleared on the display controller, insert the following coding into the operating system at installation time to clear interrupts when the computer is placed in execution after autoloading. Insert this coding into the space module after the REJ NOP 0 instruction. The purpose of the coding is to set each station (word 14 of PHYSTB on the thread) active; to write one word of sync (do-nothing) codes; and to clear active on each station. Therefore, turn on all stations or the routine will hang while trying to write. Writing on each station clears all interrupts for that station. If any other types of devices are included in the thread, avoid trying to write on any devices besides displays by adding coding to check for the device type specified in word 8 of PHYSTB.

```
LABEL     OP        ADDRESS             COMMENT
          EXT       CRT451
          LDQ       =XCRT451            NAME OF ANY CRT PHYSTB
          STQ*      SAVQ
          STQ*      STORQ
LOOP      LDA-      13,Q
          AND       =N$03C0             SAVE STATION NO. BITS 6-9
          ADD       =N$0410             ADD SEL STATION 1 + SET ACT BITS
          LDQ-      7,Q
          INQ       1                   DIRECTOR FUNCTION2
          NOP       0
          OUT       -1
          INQ       -2                  SET UP DATA WRITE
          LDA       =N$1616             WRITE SYNCS TO CLR INTERRUPTS IF
                                        ANY
          NOP       0
          OUT       -1
          LDQ*      STORQ
          LDA-      13,Q
          AND       =N$03C0
          ADD       =N$0020             CLEAR ACTIVE BIT
          LDQ-      7,Q
          INQ       1
          NOP       0
          OUT       -1
          LDQ*      STORQ
          LDA-      14,Q                THREAD WORD
          STA*      STORQ
          SUB*      SAVQ                CHECK FOR END OF THREAD
          SAZ       4
          LDQ*      STORQ
```

```
LABEL     OP       ADDRESS             COMMENT
          JMP*     LOOP
SAVQ      NUM      0                   TEMP STORAGE
STORQ     NUM      0                   TEMP STORAGE
```

3.6.14 1751 DRUM DRIVER

Description

The 1751 Drum Driver (DRMDRZ) provides a capability for data transfer to and from the drum as a mass memory device. Additionally, the drum driver handles the transfer of mass-memory resident programs into core as the result of SCHDLE requests.

The 1751 Drum Interface uses the Direct Storage Access bus for its input/output to provide completely buffered operation. The Drum Driver complements this capability by requiring control only upon end-of-operation of error condition as indicated by an interrupt.

Installation Requirements

Mass Memory: None.

Core Memory:

Driver	272 words
System tables and parameters	
Logical unit tables	3 words
Diagnostic time-out (DGNTAB)	1 word
Physical equipment table	38 words
Interrupt response routine	3 words
OVRLAY subroutine	8 words
Interrupt trap region	4 words
Diagnostic subroutine (DMDIAG)	33 words
Total	362 words

Installation Procedures

1. Insert the four-word interrupt entry which is associated with the drum interrupt line. It must contain the following, with x as the particular interrrupt line to be used.

```
                    LINEx     NUM       0
                              RTJ-      ($FE)
                              NUM       10
                              ADC       EPROC
```

2. Into LOG1A, enter:

```
                              ADC       DRUM
```

3. Enter into LOG1:

```
                              ADC       0
```

4. Into LOG2, enter:

```
                              NUM       $FFFF
```

5. Insert in SYSBUF:

```
                              ENT       INTDRUM
                              EXT       DRMINT, DRMCON, DRMERR
```

6. Insert the interrupt response routine in SYSBUF:

```
                    INTDRM    LDQ       =XDRUM
                              JMP*      (DRUM+2)
```

7. Add the following to SYSBUF:

```
                              EQU       E(2)
```

8. When adding the drum PHYSTB, the driver Priority Level is 10, the equipment code is 2, the coding is as follows:

```
WORD        LABEL       OP        ADDRESS
 0          DRUM        NUM       $12AA
 1                      ADC       DRMINT
 2                      ADC       DRMCON
 3                      ADC       DRMERR
 4-6                    NUM       -1, 0, 0
 7-9                    ADC       E*$80+1, $1066, $200
10-12                   ADC       0, 0, 0
13-18                   BSS       (6)
19-20                   ADC       0, E*$80+$8
21-22                   ADC       0, E*$80+$A
```

```
WORD     LABEL     OP      ADDRESS
23-24              ADC     0, E*$80+$C
25-26              ADC     0, E*$80*$E
27-28              ADC     8, E*$80+$1
29-30              ADC     0,0
31-32              ADC     0,-1
33-35              NUM     4,0,0
36-37              ADC     E*$80, E*$80+$4
```

9. Add the Drum Overlay Subroutine in SYSBUF:

```
         OVRLAY    0       0
                   IIN     0
                   LDA*    OVRLAY
                   ADD-    $32
                   STA*    ORVL1
                   RTJ-    ($F4)
         OVRL1     0       0
                   JMP-    ($EA)
```

10. If time-out surveillance is desired, add the following entry to the DGNTAB:

```
         LABEL     OP      ADDRESS
                   ADC     DRUM
```

11. DMDIAG, the drum diagnostic subroutine, resides in SYSBUF and handles all error recovery for the driver. If the user wishes to supply his own routine, that routine must comply with the following which apply to the present standard version:

If a read of a system directory program into allocated core produces an error, this allocated core must be released.

It can take diagnostic action as desired.

It is a closed subroutine which is entered by a RTJ instruction.

It must be entered with the drum PHYSTB address in the I register to allow access to the request parameters for diagnostic action.

The standard version types the message: MASS MEM ERR code.

Code	Significance
0	Time-out error; no completion interrupt occurred as a result of initiation of a drum operation; 1751 malfunction.
1	Internal or external reject occurred on an INP or OUT instruction. Possible causes are: equipment turned off, erroneous equipment code, or 1751 malfunction.

Code	Significance
2	Request not successfully completed because of occurrence of an irrecoverable error condition, defined previously.
3-6	Not used.
7	External reject occurred on an OUT instruction and no timing synchronizat: error was present (1751 malfunction).
8	The request completion address parameter C lies within the range of a transfer not completed because of an irrecoverable error. This condition normally occurs as the result of a SCHDLE OVRLAY request, and DMDIAG is responsible for releasing core assigned by the SPACE request processor, if desired.
9	Guarded address error on a WRITE or FWRITE request.
10	Timing synchronization error occurred while the drum was busy.
11	Timing synchronization error occurred while the drum was busy.

The nature of the error is also indicated in the Q register upon entry to DMDIAG with one of the cod(above. When DMDIAG is finished, it returns control to the driver with the I register intact.

3.6.15 1726-405 CARD READER DRIVER

Description

The 1726-405 card reader driver allows data input from the 405 card reader to core.

This driver may be installed on mass memory or in core resident.

Core Resident Installation

Installation Requirements: The core memory requirements are the length of the driver to be used plus the 110 words in SYSBUF.

SYSBUF requirements

System tables and parameters	3
Physical device table	106
Diagnostic timer table	1
	110

Driver requirements

Hardware conversion non-buffered	366
Hardware conversion buffered	387
Software conversion non-buffered	467
Software conversion buffered	488

Installation Procedures: CR405 is the COSY deck name.

1. Insert an interrupt entry similar to the following into the appropriate interrupt trap area of LOCORE (priority 8 is used for this example):

```
LABEL     OP      ADDRESS                    COMMENTS

LINEx     NUM     0
          RTJ-    ($FE)
          NUM     8
          ADC     EPROC
```

2. Insert in LOG1A after EQU Lx(*)

```
          ADC     CR405
```

3. Insert in LOG1:

```
          ADC     0
```

4. Insert in LOG2:

```
          NUM     $FFFF
```

5. If the system has a timer package, insert in the diagnostic timer table:

```
          ADC     CR405
```

6. Construct a PHYSTB for the 1726-405 using the following instructions and sample PHYSTB as a guideline:

   a. Declare the driver entry point names as external.

   b. Word 0: select the priority level of the scheduler request so that it corresponds to the priority level selected in the appropriate interrupt trap area of LOCORE (step 1). The sample PHYSTB below uses priority level 8.

   c. Words 1, 2, 3: determine the addresses of the initiator, continuator and the error routine.

   d. Word 7: select the hardware connect address. The sample PHYSTB which follows uses 0201 which is derived from using equipment number 4 and director function 1. (III.1.2.12)

```
WORD   LABEL     OP     ADDRESS                   COMMENTS

                 EXT    IN1726,CN1726,EX1726
0      CR405     NUM    $1208                     1726 CRD ENTRY
1                ADC    IN1726                    INITIATOR ENTRY
2                ADC    CN1726                    CONTINUATOR ENTRY
3                ADC    EX1726                    HANG UP ENTRY
4                NUM    -1                        DIAGNOSTIC CLOCK
5                NUM    0                         LOGICAL UNIT
```

WORD	LABEL	OP	ADDRESS	COMMENTS
6		NUM	0	CALL PARAMETER LIST
7		NUM	$0201	HARDWARE ADDRESS
8		NUM	$1972	REQUEST STATUS
9		NUM	0	STATUS WORD NO. 1
10		NUM	0	CURRENT DATA STORAGE LOCATION
11		NUM	0	LAST DATA STORAGE LOCATION
12		NUM	0	STATUS WORD NO. 2
13		NUM	0	PACKING CYCLE ADDRESS STORAGE
14		NUM	$6	EOF CARD PATTERN
15		ADC	BF1726	STARTING ADDRESS OF 80 WORD BUFFER
16		NUM	0	CURRENT CARD BUFFER ADDRESS
17		NUM	0	SUBROUTINE RETURN ADDRESS
18		NUM	0	CARD SEQUENCE NUMBER WORD
19		NUM	0	RECORD LENGTH WORD
20		NUM	0	CHECKSUM ACCUMULATOR WORD
21		NUM	0	TEMPORARY SEQUENCE STORAGE WORD
22		NUM	0	TEMPORARY STORAGE FOR FIRST WORD READ
23		NUM	0	TEMPORARY STORAGE
24		NUM	0	HOLLERITH ERROR FLAG
25-104		BZS	BF1726(80)	80 WORD BUFFER

7. Several assembly options are available. The released version of the 1726-405 driver specifies an unbuffered system using ASCII 1963. To specify different options, change the EQU BUFER and the EQU ASCI68 COSY cards. Change these cards before assembly and then make a new binary tape. Following are the possible options:

EQU	BUFER(0)	unbuffered driver
EQU	BUFER(1)	buffered driver
EQU	ASCI68(0)	driver which converts ASCII 1963
EQU	ASCI68(1)	driver which converts ASCII 1968
EQU	ASCI68(2)	driver which converts ASCII 1968 with CDC subset

8. Install the driver under a *L statement.

Mass Storage Resident Installation

Core Memory Installation Requirements: When installing the 1726-405 on mass storage resident, the core requirement is the sum of SYSBUF and MASDRV.

SYSBUF requirements

System tables and parameters	3
PHYSTB	106
Diagnostic timer table	1
	110

MASDRV requirements

MASDRV length	97
MASDRV buffer length	17 plus longest driver

To obtain the MASDRV buffer length, add the DRIVEM macro length (17) to the length of the longest driver on the system using MASDRV. (The 1713 and the 1740-501 drivers also use MASDRV.) If the 1726-405 driver is to be this length, use one of following lengths:

Hardware conversion non-buffered (367)
Hardware conversion buffered (388)
Software conversion non-buffered (468)
Software conversion buffered (489)

114 plus longest driver

The total core requirement is 224 words plus the longest driver length.

Mass Memory Installation Requirements:

DRIVEM macro length	17
1726-405 driver length	driver length

The four possible lengths are listed above under MASDRV requirements.

17 plus driver length

Installation Procedures: The COSY deck names are: CR405, DRIVEM, MASDRV.

1. Insert into LOCORE an interrupt trap entry similar to the following:

```
LABEL   OP     ADDRESS                    COMMENTS

LINEx   NUM    0
        RTJ-   ($FE)
        LABEL       OP      ADDRESS             COMMENTS

                    NUM     $10                 PRIORITY 10 IN THIS EXAMPLE
                    ADC     EPROC
```

2. Insert in LOGIA after EQU Lx(*):

```
                    ADC     CR405
```

3. Insert in LOG1:

```
                    ADC     0
```

4. Insert in LOG2:

```
                    NUM     $FFFF
```

5. If the system has a timer package, insert in the diagnostic timer table:

```
                    ADC     CR405
```

6. Construct a PHYSTB for the 1726-405 using the following instructions and sample PHYSTB as a guideline:

   a. Declare the driver entry point name (used in the *YM statement) as external.

   b. Declare MASDRV as external.

   c. In the location preceding the PHYSTB, insert an address table constant for the ordinal name used in the *YM statement such as ADC CR405X

   d. Word 0: select the priority level of the schedular request so that it corresponds to the priority level selected in the appropriate interrupt trap area of LOCORE (step 1). Mass memory drivers using the MASDRV routine must initiate at the same priority. The minimum priority level they can use is 10.

   e. Word 1: insert the address of MASDRV as the initiator address.

   f. Words 2 and 3: insert ADC 0. DRIVEM places the continuator and error routine addresses in these locations.

   g. Word 7: select the hardware connect code. The sample PHYSTB which follows uses 0201 which is derived from using equipment number 4 and director function 1. (III.1.2.12)

   h. Word 8: insert the request status word.

```
WORD    LABEL       OP      ADDRESS             COMMENTS

                    EXT     CR405X              CARD READER ORDINAL
                    EXT     MASDRV
                    ADC     CR405X              ADDRESS OF CARD READER ORDINAL
0       CR405       NUM     $120A               1726 CARD READER ENTRY
1                   ADC     MASDRV
2                   ADC     0
3                   ADC     0
```

WORD	LABEL	OP	ADDRESS	COMMENTS
4		NUM	-1	DIAGNOSTIC CLOCK
5		NUM	0	LOGICAL UNIT
6		NUM	0	CALL PARAMETER LIST
7		NUM	$0201	HARDWARE ADDRESS
8		NUM	$1972	REQUEST STATUS
9		NUM	0	STATUS WORD NO. 1
10		NUM	0	CURRENT DATA STORAGE LOCATION
11		NUM	0	LAST DATA STORAGE LOCATION
12		NUM	0	STATUS WORD NO. 2
13		NUM	0	PACKING CYCLE ADDRESS STORAGE
14		NUM	$6	EOF CARD PATTERN
15		ADC	BF1726	STARTING ADDRESS OF 80 WORD BUFFER
16		NUM	0	CURRENT CARD BUFFER ADDRESS
17		NUM	0	SUBROUTINE RETURN ADDRESS
18		NUM	0	CARD SEQUENCE NUMBER WORD
19		NUM	0	RECORD LENGTH WORD
20		NUM	0	CHECKSUM ACCUMULATOR WORD
21		NUM	0	TEMPORARY SEQUENCE STORAGE WORD
22		NUM	0	TEMPORARY STORAGE FOR FIRST WORD READ
23		NUM	0	TEMPORARY STORAGE
24		NUM	0	HOLLERITH ERROR FLAG
25-104		BZS	BF1726 (80)	80 WORD BUFFER

7. Modify the driver.

   a. Equate the initiator, continuator, and error entry points to their relative distance from location zero in the driver. Insert these equates at the end of the driver in the following format:

```
EQU   MI1726(IN1726-MS300)   INITIATOR
EQU   MC1726(CN1726-MS300)   CONTINUATOR
EQU   ME1726(EX1726-MS300)   ERROR
```

   b. Declare the equated values as entry points in the driver.

```
ENT   MI1726, MC1726, ME1726
```

c. Declare MAS300 as external to the driver.

```
          EXT      MAS300
```

d. Insert the following at the first executable program location in the driver.

```
MS300     ADC      MAS300
```

e. After the return jump to AFNR within the driver's initiator routine, replace the jump to the dispatcher with a jump to MAS300. This jump must be a one word instruction.

```
LABEL     OP       ADDRESS                     COMMENTS
Before:
IN1726    STQ-     I
          RTJ-     (AFNR)
          JMP-     (ADISP)
After:
IN1726    STQ-     I
          RTJ-     (AFNR)
          JMP*     (MS300)
```

8. Modify MASDRV.

   a. Equate LNGTH to the length of the largest driver module (including the length of the DRIVEM macro) on mass memory. For example:

```
          EQU      LNGTH($180)                 1726-405 HARDWARE NON-BUFFERED
                                               PLUS MACRO DRIVEM
```

   b. Equate NMASDR to the number of drivers on mass memory plus 1. For example:

```
          EQU      NMASDR(2)
```

9. Insert the entry points of the driver which is being installed as mass memory resident into the FRONT macro. Each mass memory module contains the DRIVEM macro and the driver body. The following is the only change necessary to the macro DRIVEM. The entry points of the driver are actually the parameters of the FRONT macro.

```
          FRONT  MI1726, MC1726, ME1726
```

10 Modify the installation tape for MSOS 2.1.

   a. Add the following to the installation tape using xx as the next available ordinal:

```
*YM, CR405X, xx
```

   b. Install MASDRV as a core resident module under an *L statement.

c. Install the macro DRIVEM and the 1726-405 driver under an *M statement in the position corresponding to the assigned ordinal.

For example:

```
*M
    DRIVEM
    CR405
```

d. Delete the following statement from the standard release install tape:

```
*S, MAS300,7FFF
```

3.6.16 1740-501 LINE PRINTER DRIVER

Description

The 1740-501 line printer driver allows data output from core memory to the 501 line printer.

Core Resident Installation

Installation Requirements: The following are the core memory requirements if the 1740-501 is to be installed in core resident:

Driver	517
System tables and parameters	6
PHYSTB	18
Diagnostic timer table	2
	543 words of core memory

Installation Procedures: The COSY deck name is PRT40.

1. Insert an entry similar to the following into the appropriate interrupt trap area of LOCORE:

LABEL	OP	ADDRESS	COMMENTS
LINEx	NUM	0	ENTRY
	RTJ-	($FE)	INTERRUPT HANDLER
	NUM	10	PRIORITY LEVEL
	ADC	EPROC	INTERRUPT RESPONSE ROUTINE

2. Insert in LOG1A after EQU Lx(*):

SLO	ADC	LP501	NON-FORTRAN
FLP	ADC	LP501	FORTRAN

3. Insert in LOG1:

	ADC	$4000	NON-FORTRAN
	ADC	$4000	FORTRAN

4. Insert in LOG2:

NUM	$FFFF	NON-FORTRAN
NUM	$FFFF	FORTRAN

5. If the timer package is used, add the PHYSTB addresses to the diagnostic timer table.

LABEL	OP	ADDRESS	COMMENTS
	ADC	LP501	NON-FORTRAN
	ADC	LP501	FORTRAN

6. Construct a PHYSTB for the 1740-501 using the following instructions and sample PHYSTB as a guideline:

   a. Declare the driver entry points as external.

   b. Word 0: select the priority level of the scheduler request so that it corresponds to the priority level selected in the appropriate interrupt trap area of LOCORE (step 1). Priority 10 is used in this example.

   c. Word 1, 2, 3: insert the addresses of the driver's initiator, continuator and error routine.

   d. Word 7: insert the hardware equipment connect code. The example below uses equipment F and station 1.

   e. Word 8: insert the request status word.

   f. Word 17: insert the FORTRAN line printer logical unit number.

WORD	LABEL	OP	ADDRESS	SIGNIFICANCE
		EXT	IN501, CN501, ER501	
0	LP501	NUM	$120A	Scheduler Request at Priority 10
1		ADC	IN501	Address of Driver Initiator
2		ADC	CN501	Address of Driver Continuator
3		ADC	ER501	Address of Driver Error Routine
4		NUM	-1	Diagnostic Clock
5		NUM	0	Logical Unit Number
6		NUM	0	Call Parameter List Location
7		NUM	$781	Hardware Address
8		NUM	$2934	Request Status
9		NUM	0	Various Status Checks
10		NUM	0	Next Core Location
11		NUM	0	Last Core Location +1
12		NUM	0	Status
13		NUM	0	Odd Character Storage and Error Code

WORD	LABEL	OP	ADDRESS	SIGNIFICANCE
14		NUM	0	Line Counter Used for Page Format
15		NUM	0	Temporary Character Storage Area
16		NUM	0	Character Counter
17		ADC	FLP-LOG1A	FORTRAN Logical Unit

7. Install the driver under an *L statement.

Mass Memory Installation

Core Memory Installation Requirements:

MASDRV (including buffer)	638
System tables and parameters	6
PHYSTB	18
Diagnostic timer table	2
	664

Mass Memory Installation Requirements:

DRIVEM	17
driver	518
	535

Installation Procedures: The COSY deck names for the 1740-501 driver, DRIVEM, and the MASDRV routine are: PRT40, DRIVEM, and MASDRV.

1. Insert an entry similar to the following into the appropriate interrupt trap area of LOCORE:

LABEL	OP	ADDRESS	COMMENTS
LINEx	NUM	0	ENTRY
	RTJ-	($FE)	INTERRUPT HANDLER
	NUM	10	PRIORITY LEVEL
	ADC	EPROC	INTERRUPT RESPONSE ROUTINE

2. Insert in LOG1A after EQU Lx(*):

SLO	ADC	LP501	NON-FORTRAN
FLP	ADC	LP501	FORTRAN

3. Insert in LOG1:

	ADC	$4000	NON-FORTRAN
	ADC	$4000	FORTRAN

4. Insert in LOG2:

```
          NUM    $FFFF          NON-FORTRAN
          NUM    $FFFF          FORTRAN
```

5. If the timer package is used, add the PHYSTB addresses to the diagnostic time table:

```
          ADC    LP501          NON-FORTRAN
          ADC    LP501          FORTRAN
```

6. Construct a PHYSTB for the 1740-501 using the following instructions and sample PHYSTB as a guideline:

   a. Declare the name used in the *YM statement as external.

   b. Declare MASDRV as external.

   c. Insert an address table constant (for the ordinal name used in the *YM statement) in the location preceding the PHYSTB such as ADC LP501M.

   d. Word 0: select the priority level of the scheduler request so that it corresponds to the priority level selected in the appropriate interrupt trap area of LOCORE (step 1). Mass memory resident drivers using the MASDRV routine must initiate at the same priority. The minimum priority level they can use is 10.

   e. Word 1: insert the address of MASDRV as the initiator address.

   f. Words 2 and 3: insert ADC 0. DRIVEM places the continuator and error routine addresses in these locations.

   g. Word 7: insert the hardware equipment connect code. The example below uses equipment F and station 1.

   h. Word 8: insert the request status word.

   i. Word 17: insert the FORTRAN line printer logical unit number.

WORD	LABEL	OP	ADDRESS	SIGNIFICANCE
		EXT	LP501M	Line Printer Ordinal
		EXT	MASDRV	
		ADC	LP501M	
0	LP501	NUM	$120A	Scheduler Request at Priority A
1		ADC	MASDRV	
2		ADC	0	
3		ADC	0	
4		NUM	-1	Diagnostic Clock
5		NUM	0	Logical Unit Number
6		NUM	0	Call Paramet er List Location
7		NUM	$781	Hardware Address

WORD	LABEL	OP	ADDRESS	SIGNIFICANCE
8		NUM	$2934	Request Status
9		NUM	0	Various Status Checks
10		NUM	0	Next Core Location
11		NUM	0	Last Core Location +1
12		NUM	0	Status
13		NUM	0	Odd Character Storage and Error Code
14		NUM	0	Line Counter used with Page Format Control
15		NUM	0	Temporary Character Storage
16		NUM	0	Character Counter
17		ADC	FLP-LOG1A	FORTRAN Logical Unit

7. Equate the initiator, continuator and error entry points to their relative distance from location zero in the driver. Insert these equates at the end of the driver.

LABEL	OP	ADDRESS	COMMENTS
	EQU	MI501(IN501-MS300)	INITIATOR
	EQU	MC501(CN501-MS300)	CONTINUATOR
	EQU	ME501(ER501-MS300)	ERROR ROUTINE

8. Declare the equated values as entry points to the driver.

```
            ENT    MI501,MC501,ME501
```

9. Declare MAS300 as external to the driver.

```
            EXT    MAS300
```

10. Insert the following at the first executable program location in the driver.

```
MS300       ADC    MAS300
```

11. After the return jump to AFNR within the driver's initiator routine, replace the jump to the dispatcher with a jump to MAS300. This jump must be a one word instruction.

LABEL	OP	ADDRESS	COMMENTS
Before:			
IN501	STQ-	I	
	RTJ-	(AFNR)	
	JMP-	(DISPAD)	
After:			
IN501	STQ-	I	
	RTJ-	(AFNR)	
	JMP*	(MS300)	

12. Modify MASDRV.

    a. Equate LNGTH to the length of the largest driver module (including the length of the macro DRIVEM) on mass memory.

       For example:

```
LABEL   OP     ADDRESS                          COMMENTS

        EQU    LNGTH($220)
```

    b. Equate NMASDR to the number of drivers on mass memory plus 1. For example:

```
        EQU    NMASDRV(2)
```

13. Insert the equated entry point names of the 1740-501 line printer driver into the FRONT macro. Each mass memory module contains the macro DRIVEM and the driver body. The following is the only change necessary to the DRIVEM macro. The entry points of the driver are actually the parameters of the FRONT macro.

```
        FRONT  MI501,MC501,ME501
```

14. Modify the MSOS 2.1 installation tape.

    a. Add the following to the installation tape using xx as the next available ordinal:

       *YM,LP501M,xx

    b. Install MASDRV as a core resident module under an *L statement.

```
*L      MASDRV

        MASDRV
```

    c. When using MASDRV, remove the *S,MAS300,7FFF statement from the installation tape.

    d. Install the DRIVEM macro and the driver under an *M statement in the position corresponding to the assigned ordinal.

```
*M                                   (xxth *M statement; step 14a)

     DRIVEM
     PRT40
```

3.7 ADDING A USER REQUEST MODULE

3.7.1 PROCEDURES

The 1700 Operating System allows 30 request processors in the standard release. The first 20 of these (T1-T20) are reserved for the Operating System. The last 10 may be designated by the user (T21-T30).

Add a request processor to the system by supplying a processing program for the request and assigning an entry point name of T21 to T30 to it. Include this program in the System Load as a core resident entry and remove the *S which links it to $7FFF. The request processor to be added to the system must adhere to the following restrictions:

The entry point name must be one from T21 to T30

Enter the request processor program by entering the location of the parameter list into the A register

The request processor must exit with a jump to the request exit entry point REQXT.

3.6.17 1777 PAPER TAPE STATION

Description

The 1777 paper tape station driver drives either:

the 1777 paper tape station or,

the 1721/1722 paper tape reader or,

the 1723/1724 paper tape punch or,

both the 1721/1722 paper tape reader and the 1723/1724 paper tape punch.

The 1777 is composed of two drivers: the 1777 paper tape station punch and the 1777 paper tape station reader.

Limitations

1704: The 1777 paper tape station driver is on equipment 1 and interrupt line 1 if it is used to drive the 1721/1722 paper tape reader and/or the 1723/1724 paper tape punch. However, because of the low speed common synchronizer, the 1777 paper tape station driver cannot be on equipment 1 if it drives the 1777.

1774 S.C.: When the 1777 paper tape station is used, there are no unique interrupt line and equipment number restrictions.

General 1777 Paper Tape Station Procedures

The 1777 paper tape station driver is either mass memory or core resident, depending on equate MASMEM.

Mass Memory Installation:

1. Load the reader and punch drivers under separate *M statements according to their ordinal. The LIBEDT *S statement is not needed to set request priority for these two modules.
2. Names which are associated with the reader and punch ordinals are:

   reader: *YM, TR1777, ordinal

   punch: *YM, TP1777, ordinal
3. Install MASDRV as a core resident module under a *L statement.
   a. Equate the length of the largest driver module on mass memory.

      EQU LENGTH( )

   b. Equate the number of drivers on mass memory plus 1.

      EQU NMASDR( )

4. Delete the statement *S, MAS300, 7FFF from the standard release installation tape.

Core Resident Installation:

1. Load the 1777 station reader and punch drivers under the *L drivers statement.
2. Load STCK under the *L DRCORE. STCK (status check) is the program containing all common routines for the 1777 station reader and punch drivers. It is used when the 1777 station reader and punch drivers are core resident. When the 1777 station reader and punch drivers are mass memory, STCK is included in each program length is not important.

1777 Paper Tape Station Reader Driver

Description: The 1777 paper tape station reader driver allows data input from the paper tape reader to core memory. The reader driver is re-entrant so that it can handle multiple readers.

Installation Requirements:

Mass Memory

driver	261
logical unit tables	3
diagnostics timer table	1
physical equipment table	20
	285 words of mass memory

Core Memory

driver	227
logical unit tables	3
diagnostic timer table	1
physical equipment table	20
	251 words of core memory

Mass Memory Procedures: The following installation procedures are unique to the 1777 paper tape station reader driver. These procedures are used to replace the 1721/1722 paper tape reader.

1. Equate MASMEM to 1 so that the 1777 paper tape station reader driver will assemble for mass memory.

```
LABEL     OP      ADDRESS
          EQU     MASMEM(1)
```

2. The following example is a four word interrupt entry associated with the low speed I/O common synchronizer. The example is assigned to line 1, priority level 10.

```
LINE1     NUM     0
          RTJ-    ($FE)
          NUM     10
          ADC     EPROC
```

3. Insert in LOGIA:

```
          ADC     PPTRDR
```

4. Insert in LOG1:

```
          ADC     0
```

5. Insert in LOG2:

```
          NUM     $FFFF
```

6. Enter as an external in the PHYSTB:

```
          EXT     MASDRV,TR1777
```

7. Add the PHYSTB to the system tables and parameters using the following coding. In this example the priority level is 10; the equipment type is 1; the equipment class is 4.

```
WORD      LABEL     OP      ADDRESS     COMMENTS
                    ADC     TR1777
0         PPTRDR    NUM     $120A
1                   ADC     MASDRV
2                   ADC     0
3                   ADC     0
4-6                 NUM     -1,0,0
7                   NUM     $A1         This can vary according to equipment
                                        number.
8                   NUM     $2012
9-19                BZS     (11)
```

8. Add the following entry to the diagnostic timer table (DGNTAB) if timeout surveillance over reader operation is desired:

```
          ADC     PPTRDR
```

9. Modify MASKT according to instructions in Part III, Section 1.2.3.

Core Memory Procedures: For the core memory installation of the 1777 paper tape station reader driver, follow the same procedures as for mass memory with the following exceptions:

1. Equate MASMEM to 0:

LABEL	OP	ADDRESS
	EQU	MASMEM(0)

2. Delete ADC TR1777 from the FHYSTB.
3. Change words 1, 2, and 3 of the PHYSTB to:

```
          ADC     PREADI
          ADC     PTREAD
          ADC     PTRERR
```

4. Replace the mass memory externals with:

```
          EXT     PREADI,PTREAD,PTRERR
```

1777 Paper Tape Station Punch Driver

Description: The 1777 paper tape station punch driver allows data output from core memory to the paper tape punch. The driver punches eight level tape only. The punch driver is re-entrant so that it can handle multiple punches.

Installation Requirements:

Mass Memory

driver	246
logical unit tables	3
diagnostic timer table	1
physical equipment table	16
	266 words of mass memory

Core Memory

driver	164
logical unit table	3
diagnostic timer table	1
physical equipment table	16
	184 words of core memory

Mass Memory Procedures: The following installation procedures are unique to the 1777 paper tape station punch driver. These procedures are used to replace the 1723/1724 paper tape punch driver.

1. Equate MASMEM to 1 so that the 1777 paper tape station punch driver will assemble for mass memory.

```
LABEL       OP        ADDRESS
            EQU       MASMEM(1)
```

2. The following example is a four word interrupt entry associated with the low speed I/O common synchronizer. The example is assigned to line 1, priority level 10.

```
LINE1       NUM       0
            RTJ-      ($FE)
            NUM       10
            ADC       EPROC
```

3. Into LOG1A enter:

```
            ADC       PPTPCH
```

4. Into LOG1 enter:

```
            ADC       0
```

5. Into LOG2 enter:

```
            NUM       $FFFF
```

6. Enter as an external in the PHYSTB:

```
            EXT       MASDRV,TP1777
```

7. Insert the following PHYSTB. The driver priority level is 10; the equipment type is 2; the equipment class is 4.

WORD	LABEL	OP	ADDRESS	COMMENTS
		ADC	TP1777	
0	PPTPCH	NUM	$120A	
1		ADC	MASDRV	
2		ADC	0	
3		ADC	0	
4-6		NUM	-1,0,0	
7		ADC	$C1	Can vary according to equipment number.
8		ADC	$2024	
9-19		BZS	(11)	

8. If time out surveillance over punch operation is desired, insert the following entry into the diagnostic timer table:

```
ADC     PPTPCH
```

9. Modify MASKT according to instructions in Part III, Section 1.2.3.

Core Memory Procedures: For the core memory installation of the 1777 paper tape station punch driver use the same procedures as for mass memory with the following exceptions:

1. Equate MASMEM to 0:

```
EQU     MASMEM(0)
```

2. Delete ADC TP1777 from the PHYSTB.
3. Change words 1, 2, and 3 of the PHYSTB to:

```
ADC     PUNCHI
ADC     PUNCDR
ADC     PCHERR
```

4. Replace the mass memory externals with

```
EXT     PUNCHI,PUNCDR,PCHERR
```

Validation Option Procedure: If the validation check and repunch is required, equate VALERR to 1 in the driver source, and assemble the driver. This is a hardware feature which is only available on the 1777 paper tape station.

```
EQU     VALERR(1)
```

3.7.2 CALLING SEQUENCE

A typical calling sequence to the request module is:

OP	ADDRESS	SIGNIFICANCE
RTJ-	($F4)	Go to monitor
NUM	$HHHH	Request code: bits 14-09 contain the processor Request number which is contained in the entry point name (T21-T30).

3.8 BUILDING AN INITIALIZER

3.8.1 AVAILABLE MODULES

CONTRL[†]	Control module
LIB[†]	Library generation module
IDRIV[†]	Input control module (input device driver)
MDRIV[†]	Mass storage driver control module
CDRIV[†]	Comment control module (comment device driver)
ILOAD[†]	Resident loader
I1[†]	Pre-resident load initialization
I2[†]	Post-resident load initialization
MSDISK[†††]	Pre-resident initialization 853/854 disk driver
MSDRUM[†††]	Pre-resident initialization 1751 drum driver
I2DISK[†††]	Post-resident initialization 853/854 disk driver
I2DRUM[†††]	Post-resident initialization 1751 drum driver
MTIDRV[††]	601 magnetic tape driver
PTIDRV[††]	1721/1722 paper tape reader driver
LPRINT[††]	1742 line printer driver

3.8.2 PROCEDURES FOR GENERATING AN INITIALIZER

1. Obtain all necessary and optional modules from MSOS COSY tape, and assemble them.
2. Use the relocatable binaries received as input to LIBEDT.
3. Assign input to the logical unit containing the binaries.

[†] Required modules

[††] Optional according to configuration

[†††] Normally either the disk or drum drivers are used, not both

4. Enter on the teletypewriter:

   Type: LIBEDT

   Press: CARRIAGE RETURN

   Type: *K, Ilu

   Press: CARRIAGE RETURN

   Type: *P

   Press: CARRIAGE RETURN

5. When all input has been read, the following appears on the teletypewriter:

   Message: LUluFAILED 02

   Type: CU

   Press: CARRIAGE RETURN

6. At this time, the absolutized binary is punched on the paper tape. The following is printed on the list device as unlinked. This is to provide easy linking of user modules when required.

   I3

   I4

7. For tape format, see Part II.4.3.1. Record 1 of the initializer tape is the absolutized checksum loader. Record 2 of the initializer is the absolutized binary programs. Therefore, when the binaries have been absolutized, insert them after the checksum loader which has been assembled and absolutized.

ABSOLUTIZED	G	ABSOLUTIZED BINARY
CHECKSUM	A	
LOADER	P	INITIALIZER

3.9 SYSTEM CHECKOUT

3.9.1 REQUIREMENTS

Hardware and memory requirements are listed under I. 1. 7. 6.

3.9.2 INSTALLATION PROCEDURES

Loading during Initialization

1. Assign the next two available mass memory system directory ordinals to SYSCOP and SYSSEG (under which the System Checkout programs will be loaded). As an example, 19 and 20 are used as system ordinals for SYSCOP and SYSSEG in this section.

   ```
   *YM, SYSCOP, 19, SYSSEG, 20
   ```

2. Load the SYSCOP program after the 19th *M; load CO1ST, CO2ND, CO3RD, and COLAST after the 20th *M.

3. Set COBOPS and/or COBOPL as entry points.

   COBOPS is the starting sector of a block of mass memory on which the COBOP program will write the failed image. If unpatched, COBOPS is assumed to be $3D29 which allows a 32K image to be written on the highest sectors of an 853 disk. Because this block may be part of scratch, do not use scratch while SYSCOP is running or if the image is to be saved.

   COBOPL is the length of transfer by COBOP. Set it according to the core size of the system.

System Size	COBOPL
16K	$3FFF
20K	$4FFF
24K	$5FFF
28K	$6FFF
32K	$7FFF assumed if unpatched

   Set COBOPS and/or COBOPL entry points in one of the following ways.

   Use a core resident program (*L program) which was loaded prior to SYSCOP, CO1ST, CO2ND, CO3RD, and COLAST,

   or

   use the *S statement. Using the *S statement, the following example reflects a 28K system with the failed image to be written starting at sector $2000:

   ```
   *S, COBOPL, 6FFF
   *S, COBOPS, 2000
   ```

4. Load COBOP using the following information:

   COBOP must be loaded under an *L (core resident program) before loading CO1ST, CO2ND, CO3RD, and COLAST.

   COBOP requires $3A of core memory.

   COBOP can only be used with an 853/854 disk assigned to equipment code 3 on the A-Q channel.

   Externals COBOPS and COBOPL may be unpatched.

5. Load SYSCOP using the following information:

   Load SYSCOP under the SYSCOP ordinal.

   SYSCOP requires $17E locations of allocatable core during execution.

   COBOPS may be unpatched.

   Schedule the SYSCOP ordinal at priority level 3 (MIPRO' may be used) in step 7 of 3.9.4.

6. Load CO1ST, CO2ND, CO3RD, and COLAST considering the following:

   Load CO1ST through COLAST under the SYSSEG ordinal so that CO1ST is first and COLAST is last.

   FMASK and/or NDISP may or may not be unpatched, depending on the configuration.

   Load SYSCOP after the following programs:

   TRVEC
   NIPROC
   NMONI
   NFNR
   DRCORE
   PARAME
   ALVOL
   NCMPRQ
   NDISP or RDISP
   COMMON
   MINT
   COBOP

7. The sample format for inserting the information specified in steps 1 through 6 is:

```
SI
*Y . . .
.
.
.
*YM. . .
.
.
.
*YM,SYSCOP,19,SYSSEG,20
.
.
.
*S, COBOPS,2000
*S, COBOPL,6FFF
.
.
.
*L
    LOCORE      0000
.
.                                          (Monitor Modules)
.
*L
.
.
.
    COBOP       hhhh
.
.
.
*M              (19th *M)
    SYSCOP      hhhh
*M              (20th *M)
    CO1ST       hhhh
    CO2ND       hhhh
    CO3RD       hhhh
    COLAST      hhhh
.
.
.
*T
```

8. After loading, use LIBEDT to set the system request priorities to four as in the following example based on the information in step 7.

   Message: J

   Type: *LIBEDT

   Press: CARRIAGE RETURN

   Message: LIB
   IN

   Type: *S,19,4,M

   Press: CARRIAGE RETURN

   Message: IN

   Type: *S,20,4,M

   Press: CARRIAGE RETURN

   Message: IN

Sample System Initializer Typeout

```
SI
*S,MAXCOR,6FFF

Q
*S,SECTOR,3E7F

Q
*S,COBOPL,6FFF

Q
*S,COBOPS,2000

Q
*I,3

Q
*C,7

Q
*V

ERROR    C

Q
*L
TIMER RJ
PP
*
MI
*LIBEDT
    LIB
    IN
*V,6
    IN
```

```
TIMER RJ
PP
*
TIMER RJ
PP
*
MI
=S019,3
SELECT OPTION          (SYSCOP)       AT LEVEL 3
```

System Initialization Printout

```
*S,ONE,7FFF
*S,TWO,7FFF
*S,THREE,7FFF
*YM,LOADSD,1,JOBENT,2,JOBPRO, 3, JPLOAD, 4, JPST, 5
*YM,JPCHGE,6,JBKILL,7,JPT13,8,MIPRO,9,LIBEDT,10
*YM,MOD1,11,MOD2,12,MOD3,13,MOD4,14,RESTOR,15
*YM,ODEBUG,16,RCOVER,17,BRKPT,18
*YM,SYSCOP,19,SYSSEG,20
*L      LOCORE
   LOCORE   0000
   SYSBUF   01D7
   SCHEDU   05F4
   NDISP    0693
   NCMPRQ   06CF
   NFNR     0700
   ADEV     076A
*M      LOADER
   LOAD     0001
   BRANCH   0001
   LIDRIV   0001
   LCDRIV   0001
   LMDRIV   0001
   LLDRIV   0001
   SCAN     0001
   CHPU     0001
   ADJOVF   0001
   CONVRT   0001
   TABSCH   0001
   TABSTR   0001
   LSTOUT   0001
   LINK1    0001
   LINK2    0001
   COREXT   0001
   DPRADD   0001
   LOADER   0001
   NAMPRO   0001
   RBDBZS   0001
   ENTEXT   0001
   XFRPRO   0001
```

```
    HEXPRO    0001
    EOLPRO    0001
    ADRPRO    0001
*L            DRCORE
    DRCORE    08AB
    ALCORE    09E0
    ALVOL     0A89
    OFVOL     0AA6
    TRVEC     0AB2
    PARAME    0ACF
    COMMON    0B2D
    NIPROC    0B44
    NEPROC    0BC0
    NMONI     0C24
    RW        0C66
    MAKQ      0D02
    MINT      0D25
    COBOP     0DD9
*M            JOBENT
    JOBENT    0021
    T11       0021
    T7        0021
    T3        0021
*M            JOBPRO
    JOBPRO    0025
    PROTEC    0025
    T5        0025
*M            JPLOAD
    JPLOAD    0033
*M            JPST
    JPST      0038
*M            JPCHGE
    JPCHGE    003A
    ASCHEX    003A
*M            JBKILL
    JBKILL    003E
*M            JPT13
    JPT13     0040
    T13       0040
*M            MIPRO
    MIPRO     0046
*M            LIBEDT
    LIBEDT    0049
*M            UTILIB
    UTILIB    0054
*M            PLINSN
    PLINSN    0061
*M            FILE
    FILE      006E
*M            GENLIB
    GENLIB    007C
```

```
*M          RESTORE DEVICE
  RESTOR    0082
*M          ODEBUG
  ODEBUG    0085
*M          RCOVER
  RCOVER    009A
  OUTSEL    009A
  DMPCOR    009A
  MASDMP    009A
*M          BRKPT
  BRKPTD    00A3
  SIFT      00A3
  BIASCI    00A3
  RETJMP    00A3
  JUMPTO    00A3
  ENTER     00A3
  ENTCOR    00A3
  PRTREG    00A3
  TERMIN    00A3
  RESUME    00A3
  DMPCOR    00A3
  MASDMP    00A3
  SETBR     00A3
  SETBRP    00A3
*M          SYS CHECKOUT
  SYSCOP    00B1
*M          SYS CHECKOUT
  CO1ST     00B5
  CO2ND     00B5
  CO3RD     00B5
  COLAST    00B5
*L          DRIVERS
                 SUBPROGRAMS WITH THE FOLLOWING ENTRY POINT NAMES HAVE NOT
                                BEEN LOADED DURING *M LOAD.

  FMASK

  DR1728    0E13
  CD1729    1170
  PTREAD    1336
  PUNCDR    140E
  TELTYP    14DD
  TAPEDR    161D
  FRWA      1753
  FRWB      180D
  RECOVT    18DF
  TAPE      1951
```

```
  CARDRD   195B
  PRINTR   1AC0
  DISKWD   109E
  SPACE    1E48
*S,TIMINT,7FFF
*S,SNAPE,7FFF
*S,PARITY,7FFF
*S,IPROC1,7FFF
*S,T30,7FFF
*S,T29,7FFF
*S,T28,7FFF
*S,T27,7FFF
*S,T26,7FFF
*S,T25,7FFF
*S,T24,7FFF
*S,T23,7FFF
*S,T22,7FFF
*S,T21,7FFF
*S,T20,7FFF
*S,T19,7FFF
*S,T18,7FFF
*S,T17,7FFF
*S,T16,7FFF
*S,T13,7FFF
*S,T11,7FFF
*S,T8,7FFF
*S,T7,7FFE
*S,T5,7FFF
*S,T3,7FFF
*S,JKIL,7FFF
*S,RWBA,7FFF
*S,RW609,7FFF
*S,DEBUG,7FFF
*S,DTIMER,7FFF
*S,MAS300,7FFF
*T
0122

IN

*S,7,4,M
IN

*S,8,3,M
IN

*S,9,1,M
IN

*S,10,2,M
IN
```

```
*S,11,3,M
IN

*S,12,3,M
IN

*S,13,3,M
IN

*S,14,3,M
IN

*S,15,4,M
IN

*S,16,5,M
IN

*S,17,2,M
IN

*S,18,0,M
IN

*S,19,4,M
IN

*S,20,4,M
IN

*U
```

Loading after Initialization

The mass memory modules may be updated after initialization by using the *M instruction in LIBEDT. To do so, perform the following instructions during initialization.

Assign ordinals (step 1).

Set COBOPS and COBOPL entry points (step 3).

Load COBOP, since it is core resident (step 4).

Noting that both SYSCOP and COLAST must be followed by an *T if the input device is magnetic tape, continue after initialization with the following procedures.

1. Type: *LIBEDT

   Press: CARRIAGE RETURN

   Message: LIB
   IN

2. Type: *K,Ilu

   lu contains the relocatable binaries of the mass memory modules

   Press: CARRIAGE RETURN

   Message: IN

3. Type: *M,19,,M,N

   Press: CARRIAGE RETURN

   SYSCOP is loaded

   Message: IN

4. Type: *M,20,,M,N

   Press: CARRIAGE RETURN

   CO1ST, CO2ND, CO3RD, and COLAST are loaded

   Message: IN

3.9.4 USER INSTRUCTIONS

Be sure that COBOP is intact before using the following procedures to check for system status or for system malfunctioning.

1. Set the STEP/RUN switch to RUN
2. Clear all registers except the A and Q registers
3. Set the P register to the starting address of COBOP
4. Set the SELECTIVE STOP switch
5. Set the STEP/RUN swtich to RUN
6. When the machine stops, select the Q register to examine core data as explained in II.1.1.2.

   If FFFF appears on the push button register, the hardware malfunctioned. Repeat steps 1 through 6.

   If 0000 appears on the push button register, restart the system.
7. To schedule SYSCOP:

   Press: MANUAL INTERRUPT

   Message: MI

   Type: =Sxxx,3

   xxx is the ordinal number

   3 is the priority level as specified in step 5 of 3.9.2.

   Example: =S019,3

   Press: CARRIAGE RETURN
8. Message: SELECT OPTION

9. SYSCOP may be rerun as often as necessary on the same image if the area on which the failed image is written is not destroyed. Respond with one of the following five options which will remain in effect until the DUMP routine is executed:

Option	Action		Significance
1	Type:	*Z	To release package; no further input is necessary but the package may be rescheduled.
	Press:	CARRIAGE RETURN	
	Message:	FINISH SYSCOP	
2	Type:	0	to transfer control to DUMP routine
	Press:	CARRIAGE RETURN	
	Message:	DUMP	
	Type:	one of the following	
		*Z	to exit from the package
		*R	to repeat the SYSCOP package beginning with step 8
		*Dxxxx,yyyy	to dump cells xxxx to cells yyyy from the failed image; DUMP appears as a message to request the next input or if invalid data is typed
	Press:	CARRIAGE RETURN	return to step 8
3	Type:	1	to print error messages only on list logical unit; return to step 8
	Press:	CARRIAGE RETURN	
4	Type:	2	to print errors plus supporting messages on list logical unit; return to step 8
	Press:	CARRIAGE RETURN	
5.	Type:	3	to print errors and all supporting messages on list logical unit; return to step 8
	Press:	CARRIAGE RETURN	

Loading Example

The failed image has already been written on mass memory by COBOP for the following verification of correct loading after the scheduling of SYSCOP:

Typeout/Printout	Significance
SYSCOP START	output on list lu; indicates SYSCOP is scheduled and has begun
IMAGE START SECTOR IS 2000	

Typeout/Printout	Significance
SELECT OPTION	appears on comment logical unit
0	user types to transfer control to DUMP routine
DUMP	appears on comment logical unit
*R	user types to repeat the SYSCOP package
SELECT OPTION	appears on comment logical unit
1	user types to request printing of error messages
⋮ errors, if any	errors appear on list logical unit
SELECT OPTION	appears on comment logical unit
*Z	user types to release package
FINISH SYSCOP	appears on list logical unit; package is complete; core is released

Printout Examples

Option 1:

```
***SYSTEM DIRECTORY ERROR
INDEX 000F HAS INVLAID REQ PRI 0004
INDEX 0001 TOO LONG FOR REQ PRI 0000
INDEX 0014 TOO LONG FOR REQ PRI 0004
```

Option 2:

```
  A       Q       I    REGISTER
0000    0000    1FA2
PRI LVL WAS 0000
LU  04  CURRENT PARA LIST AT 21F5
RC  0800
C   0000
TH  FFFF
LU  1004
N   0010
S   208B
I/O  IN  PROGRESS
RETURN FOR  FNR WAS  14E3
RETURN FOR  CMR  WAS  1555
LAST ENTRY TO BE SCHEDULED
0360/  12AA   0D25   0364   0091
THERE WERE  0000 OF THE 0101 VOLATILE WORDS ASSIGNED
ALLOCATABLE CORE MAP
```

```
INDEX START LNGTH THRD DUMP
0002   1E56   0144   1E58   C8FE   6C22   40FF   0822   0927
000A   1F9A   041C   1F9C   C8FE   6400   0ABB   0B00   5800
EMPY   23B6   064A   FFFF   C8FE   6400   0ABC   0814   B032
***SYSTEM DIRECTORY ERROR
INDEX 000F HAS INVALID REQ PRI   0004
INDEX 0001 TOO LONG FOR REQ PRI 0000
INDEX 0014 TOO LONG FOR REQ PRI 0004
SYSTEM NOT SWAPPED
JP WAS IN CORE
FILE1   FILE 2   FILE3   FILE4   LOADR BP
1E58    1F9C     0000    0000    0000   0000
JP LOCKED OUT FOR LIBEDT OR RECOVERY
```

Option 3:

```
 A       Q       I     REGISTER
0000    0000    1FA2
MAX CORE WAS 6FFF WITH 2A00 TO 6FFE UNPROT
MAXSEC WAS    00003E7F
MAX CORE WAS  6FFF WITH 2A00 TO 6FFE UNPROT
MAXSEC WAS    00003E7F
PRI  LVL  WAS  0000
LINE    01 LAST INTERRUPTED 21FD
LINE    04 LAST INTERRUPTED 0422
LINE    05 LAST INTERRUPTED 2220
LINE    0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
LEVEL F    A  D  B  9  A  6  6  9  9  D  D  6  6  6  6
LINE       1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
LEVEL F    A  D  B  9  A  6  6  9  9  D  D  6  6  6  6
INTRPT STACK LEVEL
 -1
LU    04 CURRENT PARA LIST AT 21F5
RC    0800
C     0000
TH    FFFF
LU    1004
N     0010
S     208B
I/O  IN PROGRESS
RETURN FOR FNR WAS 14E3
RETURN FOR CMR WAS 1555
NUM OF SCHEDL STACK ENTRIES WAS 18
NUM OF SCHEDL CALLS STACKED WAS 00
LAST ENTRY TO BE SCHEDULED
0360/    12AA    0D25    0364    0091
THERE WERE 0000 OF THE 0101 VOLATILE WORDS ASSIGNED
ALLOCATABLE CORE MAP
INDEX START LNGTH THRD DUMP
0002   1E56   0144   1E58   C8FE   6C22   40FF   0822   0927
000A   1F9A   041C   1F9C   C8FE   6400   0ABB   0B00   5800
EMPY   23B6   064A   FFFF   C8FE   6400   0ABC   0814   B032
```

```
***SYSTEM DIRECTORY ERROR
INDEX 000F HAS INVALID REQ PRI   0004
INDEX 0001 TOO LONG FOR REQ PRI 0000
INDEX 0014 TOO LONG FOR REQ PRI 0004
SYSTEM NOT SWAPPED
JP WAS IN CORE
FILE1   FILE2   FILE3   FILE4   LO  DR  BP
1E58    1F9C    0000    0000    0000    0000
JP LOCKED OUT FOR LIBEDT OR RECOVERY
```

3.10 SYSTEM CONFIGURATOR

3.10.1 DESCRIPTION

The System Configurator program is an online unprotected program which provides statistics for a proposed system and generates an installable 1700 2.1 Mass Storage Operating System for a required system.

3.10.2 REQUIREMENTS

Hardware Requirements

The minimum hardware configuration is the same as for MSOS 2.1. For an optimum installation, refer to I.1.7.7.

Software Requirements

A minimum of 3000 words of unprotected core is necessary to execute SYSCON.

Logical Unit Requirements

For installation of SYSCON:

mass storage device is LUN 8

paper tape reader is LUN 2

magnetic tape drive (if present) is LUN 6

All other logical units are dependent upon installation.

Installation Requirements

Replace the MSOS system program ADRPRO with the correct version defined in PSR #560 (PSR summary #32). The corrective coding is as follows:

```
LABEL    OP      ADDRESS

ADRPRO   DCK/    I=lu, H=lu

         DEL/    94

         LDQ-    I

         END/
```

3.10.3 INSTALLATION PROCEDURES

MSOS 2.1 must be installed; the job processor must be in core.

1. Type: *LIBEDT

   Press: CARRIAGE RETURN

   Message: LIB
   IN

2. If using magnetic tape,

   a. Mount the installation magnetic tape on LU6, the magnetic tape device

   b. When READY ,

      Type: *K, I6

      Press: CARRIAGE RETURN

      Message: IN

      Type: *V, 6

      Press: CARRIAGE RETURN

      Message: IN

If using paper tape,

   a. Mount the first paper installation tape on LU2, the paper tape reader

   b. Press: CLEAR

      Type: *K, I2

      Press: CARRIAGE RETURN

      Message: IN

      Type: *V, 2

      Press: CARRIAGE RETURN

      Message: IN

3. As the tape is read, the System Configurator is installed on the program library, and the following appears on the standard print device:

```
IN

*L, CONFIG
IN

*K,I6,P8
IN

*P,F,GOCONF
     CONFIG    3489
     GOCONF    349B
     SCDKIO    349D
     ERROR     34BF
     DCTOAS    351A
     GETITM    3561
     CALADR    365A
     INCPTR    3698
     GETFLE    36B9
     GO1A      371B
     OPTCHK    372E
     INPREC    379A
     MESSGS    38FA
     SCNOPT    3998
     INITAL    3A06
     CONVRT    3AFD
     CONTRL    3B76
     CORECT    3B98
     INSINP    3C06
     SCNREC    3C1C
IN

*K,I8
IN

*N,CONF1A,,,B
IN

*K,I6
IN

*P,F,GO1B
     CONFIG    3489
     GOCONF    349B
     SCDKIO    349D
     ERROR     34BF
     DCTOAS    351A
     GETITM    3561
     CALADR    365A
     INCPTR    3698
     GETFLE    36B9
     GO1B      371B
     DEFINE    3722
     PARCHK    38E7
     PAMCHK    3916
     PARTIT    3943
     SEARCH    3963
     SCNREC    39BA
     INPREC    3A4D
     CONTRL    3BAD
     VALPRO    3BCF
     VALCHK    3C1D
     PICKUP    3D3C
IN

*K,I8
IN

*N,CONF1B,,,B
IN

*K,I6
IN

*P,F,GO1C
     CONFIG    3489
     GOCONF    349B
     SCDKIO    349D
     ERROR     34BF
     DCTOAS    351A
     GETITM    3561
     CALADR    365A
     INCPTR    3698
     GETFLE    36B9
     GO1C      371B
     SYSDAT    371F
     SYSINS    377C
     INSINP    37D6
     INCINS    37EC
     GETCHR    37FF
     STOCHR    381D
     WRTMMR    3845
     RDSKEL    387E
     INITCM    38A8
     COMMNT    38E3
IN

*K,I8
IN

*N,CONF1C,,,B
IN

*K,I6
IN
```

```
*P,F,GO1D
        CONFIG    3489
        GOCONF    349B
        SCDKIO    349D
        ERROR     34BF
        DCTOAS    351A
        GETITM    3561
        CALADR    365A
        INCPTR    3698
        GETFLE    36B9
        GO1D      371B
        SPECF1    371F
        PARCHK    38E2
        BKCMVR    3911
        SPCPAR    3970
        SEARCH    39A3
        SCNREC    39FA
        CONTRL    3A8D
        INPREC    3AAF
        CORECK    3C0F
        CORECT    3C2B
        CONVRT    3099
        INSINP    3D12
IN

*K,I8
IN

*N,CONF1D,,,B
IN

*K,I6
IN

*P,F,GO1E
        CONFIG    3489
        GOCONF    349B
        SCDKIO    349D
        ERROR     34BF
        DCTOAS    351A
        GETITM    3561
        CALADR    365A
        INCPTR    3698
        GETFLE    36B9
        GO1E      371B
        SPECF2    371F
        PAMCH2    3823
        INSURT    38DA
        INCINS    397F
        INSINP    3992
        SEARCH    39A8
        CORECT   39FF
        SCNREC   3A6D
        CONTRL   3B0C
        INPREC   3B22
        CNVTNO   3C82
        PICKUP   3CD8
IN

*K, I8
IN

*N, CONF1E,,,B
IN

*K, I6
IN

*P, F, GO1F
        CONFIG   3489
        GOCONF   3498
        SCDKIO   349D
        ERROR    34BF
        DCTOAS   351A
        GETITM   3561
        CALADR   365A
        INCPTR   3698
        GETFLE   36B9
        GO1F     371B
        VARPRO   3725
        RNGCHK   3823
        SCNREC   390E
        CORECT   39A1
        VALCHK   3A0F
        INPREC   3B2E
        CONTRL   3C8E
        CNVTNO   3CB0
        PICKUP   3D06
IN

*K, I8
IN

*N, CONF1F,,,B
IN

*K, I6
IN

*P, F, GO2
        CONFIG   3489
        GOCONF   349B
        SCDKIO   349D
        ERROR    34BF
        DCTOAS   351A
        GETITM   3561
        CALADR    365A
        INCPTR    3698
        GETFLE    36B9
        GO2       371B
        PHASE2    371F
        EQUIVA    379C
        INSERT    37D4
        DELETE    38AB
        GETVAL    38E1
        CVTNUM    3985
        GETNUM    39D8
        REDREC    39FA
        GETCHR    3A4F
        GNSCHR    3A6D
        STOCHB    3A7A
        DECASC    3AA2
        MMREAD    3B0D
        OUTREC    3B2E
        HICORE    3B3E
        INTREG    3B73
        PICKUP    3D3D
        P2NAM1    3D4D
IN

*K, I8
IN

*N, CONF2A,,,B
IN

*K, I6
IN

*P, F, GO2
        CONFIG    3489
        GOCONF    349B
        SCDKIO    349D
        ERROR     34BF
        DCTOAS    351A
        GETITM    3561
        CALADR    365A
        INCPTR    3698
        GETFLE    36B9
        GO2       371B
        PHASE2    371F
        EQUIVA    379C
        DELETE    37D4
        GETVAL    380A
        CVTNUM    38AE
        GETNUM    3901
        REDREC    3923
        GETCHR    3978
        GNSCHR    3996
        STOCHR    39A3
        DECASC   39CB
        MMREAD   3A36
        OUTREC   3A57
        MSKTBL   3A67
        FTNLVL   3B65
        SCHSTK   3BEE
        PICKUP   3C2B
        P2NAM2   3C3B
IN

*K,I8
IN

*N,CONF2B,,,B
IN

*K,I6
IN

*P,F,GO2
        CONFIG   3489
        GOCONF   349B
        SCDKIO   349D
        ERROR    34BF
        DCTOAS   351A
        GETITM   3561
        CALADR   365A
        INCPTR   3698
        GETFLE   36B9
        GO2      371B
        PHASE2   371F
        EQUIVA   379C
        DELETE   37D4
        GETVAL   380A
        CVTNUM   38AE
        GETNUM   3901
        REDREC   3923
        GETCHR   3978
        GNSCHR   3996
        STOCHR   39A3
        DECASC   39CB
        MMREAD   3A36
        OUTREC   3A57
        LUTBLS   3A67
        DGNTAB   3D06
        PICKUP   3D48
        P2NAM3   3D58
IN

*K,I8
IN

*N,CONF2C,,,B
IN
```

```
*K,I6
IN

*P,F,GO2
     CONFIG    3489
     GOCONF    349B
     SCDKIO    349D
     ERROR     34BF
     DCTOAS    351A
     GETITM    3561
     CALADR    365A
     INCPTR    3698
     GETFLE    36B9
     GO2       371B
     PHASE2    371F
     EQUIVA    379C
     INSERT    37D4
     DELETE    38AB
     GETVAL    38E1
     CVTNUM    3985
     GETNUM    39D8
     REDREC    39FA
     GETCHR    3A4F
     GNSCHR    3A6D
     STOCHR    3A7A
     DECASC    3AA2
     MMREAD    3B0D
     OUTREC    3B2E
     OUTLNN    3B3E
     FTNMSK    3BE3
     PRESET    3C0A
     PICKUP    3C73
     P2NAM4    3C83
IN

*K,I8
IN

*N,CONF2D,,,B
IN

*K,I6
IN

*P,F,GO3A
     CONFIG    3489
     GOCONF    349B
     SCDKIO    349D
     ERROR     34BF
     DCTOAS    351A
     GETITM    3561
     CALADR    365A
     INCPTR    3698
     GETFLE    36B9
```

```
         GO3A      371B
         PHASE3    371F
         PACKAG    378C
         INSPGM    37F1
         DELPGM    38CA
         OUTCRD    3905
         XTCORE    397E
         INPBIN    3992
         OUTBIN    39E6
         UNLOAD    3A43
         GETVAL    3A84
         CVTNUM    3B28
         GETNUM    3B7B
         GETCHR    3B9D
         GNSCHR    3BBB
         STOCHR    3BC8
         BINASC    3BFC
         PICKUP    3C3D
         PAGEJT    3C4D
         PRNTLN    3C5A
         PACKLN    3C7B
         NEWHDR    3CAF
         STAPGM    3CE2
         STAPCK    3D52
IN

*K,I8
IN

*N,CONF3A,,,B
IN

*K,I6
IN

*P,F,GO3B
         CONFIG    3489
         GOCONF    349B
         SCDKIO    349D
         ERROR     34BF
         DCTCAS    351A
         GETITM    3561
         CALADR    365A
         INCPTR    3698
         GETFLE    36B9
         GO3B      371B
         INPBIN    3722
         GETVAL    3776
         CVTNUM    381A
         GETNUM    386D
         GETCHR    388F
```

```
    GNSCHR   38AD
    BINASC   38BA
    PICKUP   3907
    PAGEJT   3917
    PRNTLN   3924
    PACKLN   3945
    OUTBIN   3979
    STAEND   39D6
IN

*K,I8
IN

*N,CONF3B,,,B
IN

*U
```

4. Message: IN

   Type: *Z

   Press: CARRIAGE RETURN

   Message: J

   System Configurator is installed.

3.10.4 VERIFICATION

Description

The verification procedure demonstrates that SYSCON is correctly installed. It exercises the STATISTICS and CONVERSE options of SYSCON. The following procedure fully covers an optimum as well as the minimum hardware configuration.

Requirements

MSOS must be installed, and the job processor must be in core.

Procedure

1. If using a minimum system,

   a. Mount the first paper tape containing SYSCON definitions and skeletons on LUN2

   b. When LUN2 is CLEAR, continue with step 2

   If using a non-minimum system,

   a. Mount the magnetic tape containing SYSCON definitions and skeletons on LUN6, the magnetic tape device

   b. When LUN6 is READY, continue with step 2

2. Type: *P

   Press: CARRIAGE RETURN

   Message: J

3. Type: *CONFIG

   Press: CARRIAGE RETURN

   Message: OPTIONS (STATISTICS, CONFIGURE, CONVERSE)

4. Type: ST, CONV

   Press: CARRIAGE RETURN

   Teletypewriter paper advances

5. If using a minimum system,

   Type: *J, 2

   Press: CARRIAGE RETURN

If using a non-minimum system,

Type: *J,6

Press: CARRIAGE RETURN

6. SYSCON reads in the SYSCON system definitions and skeletons tape.

7. If using a minimum system, the following message appears:

   Message: UNLOAD SYSTEM DEFINITIONS, LOAD SYSTEM SPECIFICATIONS

   Remove the system definitions and skeletons paper tape from the paper tape reader.

8. The verification data set program is on the COSY source tape under the deck name VERIFY. If a COSY source tape is not available, the verification data set program is listed in section 4.3.13 System Configurator Release Tape Format. Transfer the program to either paper tape, cards, or magnetic tape. Place the verification data set program on the input device to be used.

9. Type: *I,lun

   lun is the device to be used

   Press: CARRIAGE RETURN

10. SYSCON reads in the verification data set program (see contents of listing in II.4.3.13). After several records are read in, the following message appears:

    Message: ERROR,10,5

11. Type: *1721

    Press: CARRIAGE RETURN

12. Type: *V

    Press: CARRIAGE RETURN

    SYSCON continues to read in the verification data set program until it is completely in.

    Message: J

13. If the output is similar to that given in steps 14 and 16 or 15 and 16, SYSCON is installed correctly.

14. If the list device is other than the teletypewriter, the following is a sample of the printout which appears on the list device.

OPTIONS (STATISTICS, CONFIGURE, CONVERSE)

```
    1    ST, CONV
    2    *J, 6
    3    *I, 2
         **          VERIFICATION DECK FOR SYSCON
         **          SPECIFICATION LIST
         **
    4    *SYSTEM HARDWARE DEVICES
         **
         **     INVALID COMPONENT--USED TO VERIFY CONVERSE OPTION
    5    *+1703,
ERROR,  10,      5
         **     1723/1724 PAPER TAPE PUNCH
    6    *+1721,
    7    *V
    8    *+1723,
         **     1711/1712 TELETYPE
    9    *+1711,
         **     1738 DISK CONTROLLER WITH 853-4 DISK DRIVES
   10    *+1738/853-4,
         **
   11    *CORE RESIDENT  FOREGROUND PROGRAMS
         **
         **     E006*2.1 MONITOR PACKAGE
   12    *+MONITOR,
         **
   13    *MASS RESIDENT  FOREGROUND PROGRAMS
         **
         **     JOB  PROCESSOR WITH LOADER, LIBRARY EDIT, BREAKPOINT, RECOVERY
   14    *+JOB PROCESSOR,
         **
         **
   15    *PROGRAM LIBRARY PROGRAMS
         **
   16    *+FTN RUNTIME LIBRARY,
         **
   17    *TERMINATE
```

15. If the list device is the teletypewriter, information similar to the following appears on the teletypewriter:

```
*CONFIG
OPTIONS (STATISTICS, CONFIGURE, CONVERSE)

ST, CONV
    1    ST, CONV

*J,7
    2    *J,7

*I,2
    3    *I,2
         **           VERIFICATION  DECK  FOR  SYSCON
         **           SPECIFICATION  LIST
         **
    4    *SYSTEM  HARDWARE  DEVICES
         **
         **     INVALID COMPONENT--USED TO VERIFY CONVERSE OPTION
    5    *+1703,
ERROR,  10,       5


*+1721,
         **      1723/1724 PAPER TAPE PUNCH
    6    *+1721,
*V
    7    *V
```

16. Data similar to the following appears on the list device after the information in steps 14 or 15.

```
               SYSTEM STATISTICS

CORE RESIDENT FOREGROUND PROGRAMS

          MONITOR PROGRAMS

TRVEC        29  P      0  D      0  C
COMMON       23  P      0  D      0  C
NIPROC      124  P      0  D      0  C
NEPROC      100  P      0  D      0  C
NMONI        66  P      0  D      0  C
PARAME       94  P      0  D      0  C
ALVOL        29  P      0  D      0  C
OFVOL        12  P      0  D      0  C
NCMPRG       49  P      0  D      0  C
NFNR        106  P      0  D      0  C
MAKG         35  P      0  D      0  C
ALCORE      169  P      0  D      0  C
DRCORE      309  P      0  D      0  C
MINT        180  P      0  D      0  C
RW          156  P      0  D      0  C
ADEV        321  P      0  D      0  C
SCHEDU      159  P      0  D      0  C
NDISP        60  P      0  D      0  C
```

MASS RESIDENT FOREGROUND PROGRAMS

JOB PROCESSOR

LOAD	301	P	0	D	0	C
BRANCH	715	P	0	D	0	C
LIDRIV	94	P	0	D	0	C
LCDRIV	45	P	0	D	0	C
LMDRIV	34	P	0	D	0	C
LLDRIV	14	P	0	D	0	C
SCAN	183	P	0	D	0	C
CHPU	11	P	0	D	0	C
ADJOVF	21	P	0	D	0	C
CONVRT	24	P	0	D	0	C
TABSCH	31	P	0	D	0	C
TABSTR	41	P	0	D	0	C
LSTOUT	47	P	0	D	0	C
LINK1	29	P	0	D	0	C
LINK2	43	P	0	D	0	C
COREXT	56	P	0	D	0	C
DPRADD	23	P	0	D	0	C
LOADER	216	P	0	D	0	C
NAMPRO	219	P	0	D	0	C
RBDBZS	259	P	0	D	0	C
ENTEXT	134	P	0	D	0	C
XFRPRO	21	P	0	D	0	C
HEXPRO	266	P	0	D	0	C
EOLPRO	141	P	0	D	0	C
ADRPRO	93	P	0	D	0	C
JOBENT	95	P	0	D	0	C
T7	132	P	0	D	0	C
T11	45	P	0	D	0	C
T3	50	P	0	D	0	C
JOBPRO	351	P	0	D	0	C
PROTEC	880	P	0	D	0	C
T5	49	P	0	D	0	C
JPLOAD	387	P	0	D	0	C
JPST	111	P	0	D	0	C
JPCHGE	196	P	0	D	0	C
ASCHEX	101	P	0	D	0	C
JBKILL	106	P	0	D	0	C
JPT13	19	P	0	D	0	C
T13	480	P	0	D	0	C
RESTOR	220	P	0	D	0	C
LIBEDT	1050	P	0	D	0	C
UTILIB	1177	P	0	D	0	C
PLINSN	1227	P	0	D	0	C
FILE	1294	P	0	D	0	C

```
GENLIB          507  P          0  D          0  C
BRKPTDF         198  P          0  D          0  C
BIASCI           90  P          0  D          0  C
SIFT            141  P          0  D          0  C
RETJMP           23  P          0  D          0  C
JUMPTO           22  P          0  D          0  C
ENTER            22  P          0  D          0  C
ENTCOR           33  P          0  D          0  C
PRTREG           53  P          0  D          0  C
SETBRP          117  P          0  D          0  C
TERMIN           73  P          0  D          0  C
DMPCOR          131  P          0  D          0  C
MASDMP          241  P          0  D          0  C
RESUME          195  P          0  D          0  C
RCOVER          335  P          0  D          0  C
OUTSEL           71  P          0  D          0  C
DMPCOR          173  P          0  D          0  C
MASDMP          245  P          0  D          0  C

          MASS RESIDENT DRIVERS


CORE RESIDENT FOREGROUND PROGRAMS

          INPUT/OUTPUT DRIVERS

PTREAD          216  P          0  D          0  C
PUNODR          207  P          0  D          0  C
TELTYP          320  P          0  D          0  C
TAPE             10  P          0  D          0  C
DISK            175  P          0  D          0  C

          SPACE PROGRAM

SPACE          2982  P          0  D          0  C
```

```
          PROGRAM LIBRARY PROGRAMS

          FORTRAN RUNTIME NON-REENTRANT, NON-RUNANYWHERE LIBRARY

Q8EXPN     152  P       0  D       0  C
Q8PRMS      16  P       0  D       0  C
Q8AB        16  P       0  D       0  C
IFALT       22  P       0  D       0  C
SIGN        35  P       0  D       0  C
FXFL        96  P       0  D       0  C
FXPPRG     154  P       0  D       0  C
SQRTF      109  P       0  D       0  C
LNUPRG     114  P       0  D       0  C
TANH       106  P       0  D       0  C
SINCOS     190  P       0  D       0  C
ARCTRG     145  P       0  D       0  C
FLOAT      507  P       0  D       0  C
Q8QINI     202  P       0  D       0  C
Q8QEND      20  P       0  D       0  C
Q8OMP      187  P       0  D       0  C
Q8RWBU     277  P       0  D       0  C
Q8ERRM     211  P       0  D       0  C
Q8DFIO     174  P       0  D       0  C
Q8QX        79  P       0  D       0  C
Q8QUNI      88  P       0  D       0  C
Q8FGET     111  P       0  D       0  C
Q8MAGT      63  P       0  D       0  C
TAPCON     159  P       0  D       0  C
TOCK        28  P       0  D       0  C
PSSTOP      55  P       0  D       0  C
Q8PAND      85  P       0  D       0  C
Q8EXP9     173  P       0  D       0  C
Q8EXP1     123  P       0  D       0  C
Q8IFRM      63  P       0  D       0  C
Q8FS       472  P       0  D       0  C
Q8TRAN    1729  P       0  D       0  C

          CORE MEMORY MAP

CORE RESIDENT           2963
ALLOCATABLE             2968
UNPROTECTED            26837
SYSTEM COMMON              0
NON-SYSTEM                 0
```

3.10.6 INSTALLATION OF MSOS 2.1 GENERATED BY THE SYSTEM CONFIGURATOR

After specifying and configuring a system (3.10.3), use the system initializer to install the new system with the system installation programs and the relocatable binary SYSDAT programs.

Minimum System

If the only input device is a paper tape reader, the installation procedures are basically the same as those described in II. 2. 1 and II. 2. 2.

1. Load and execute the system initializer as in II. 2. 1.

2. Continue installing using the procedures in II. 2. 2, replacing step 3 of these procedures with:

   Type: *S, MAXSEC, xxxx

   Press: CARRIAGE RETURN

   Message: Q

3. When the system initializer types Q in step 7, mark the leader of tape under the paper tape reader and remove the paper tape.

4. Mount the relocatable binary SYSDAT program paper tape in the paper tape reader.

5. Type: *V

   Press: CARRIAGE RETURN

   Message: LUN, lun, FAILED
   ACTION

   The paper tape reader is out of tape; mount the paper tape containing the system installation programs placing the leader marked in step 3 under the reader.

6. Type: RP

   Press: CARRIAGE RETURN

7. Continue with the installation as described in step 8 of II. 2. 2.

Non-Minimum System

Use the following instructions for a system containing a paper tape reader and another device.

1. Load and execute the system initializer as described in II. 2. 1.

2. Load input device a with the system installation programs.

3. Load input device b with the relocatable binary of the SYSDAT program.

4. Continue installing using the instructions in II. 2. 2 replacing step 3 of these procedures with:

   Type: *S, MAXSEC, xxxx

   Press: CARRIAGE RETURN

   Message: Q

5. Use the sytem initializer to assign the output and comment devices as discussed in II. 2. 2, step 6.

6. Assign input device a (containing system installation programs) as the input device.

   Type: *I, a

   Press: CARRIAGE RETURN

   Message: Q

7. Type: *V

   Press: CARRIAGE RETURN

   Message: Q

8. Assign input device b (containing the relocatable binary of the SYSDAT program) as the input device.

   Type: *I, b

   Press: CARRIAGE RETURN

   Message: Q

9. Type: *

   Press: CARRIAGE RETURN

10. The relocatable binary of the SYSDAT program is loaded into the system.

    Message: ACTION

11. Type: CU

    Press: CARRIAGE RETURN

12. Assign input device a as the input device.

    Type: *I, a

    Press: CARRIAGE RETURN

    Message: Q

13. Type: *

    Press: CARRIAGE RETURN

14. Continue the installation with step 8 of II. 2. 2.

RELEASE RELATED AIDS 4

4.1 CONVENTIONS

1. Terminate each teletype input by pressing CARRIAGE RETURN.

2. To erase a teletype line:

   a. Type: RUB OUT, LINE FEED

   b. Press: CARRIAGE RETURN

3. After mounting a paper tape on the paper tape reader, press READY MASTER CLEAR on the reader.

4. If the teletype BREAK light is on, press BREAK RELEASE on the teletypewriter before attempting to type.

5. Core locations are base 16; lengths are base ten.

6. When using the 1713 Teletypewriter, it must be in K mode.

4.2 EQUIPMENT ASSIGNMENTS

4.2.1 LOGICAL UNIT, EQUIPMENT, AND INTERRUPT LINE ASSIGNMENTS

The released system configuration is:

Lun	Device	Equipment Number	Interrupt Line
1	Core Allocator	(Unavailable to operator)	
2	1721/1722	slow speed	1
3	1723/1724	slow speed	1
4	1711/1712/1713	slow speed	1
5	1729	slow speed	1
6,7	1731-601	7	3 (A/Q channel)
8	1738-853/854	3	4
9	1742	F	5
10	Dummy		
11	1728-430	8	10
12	1729-2	C	11

4.2.2 INITIALIZER LOGICAL UNIT AND EQUIPMENT ASSIGNMENTS

The lu numbers which are preceded by asterisks refer to the devices which are preset to be the standard devices during the execution of the initializer until the TIMER RJ message appears.

Lun	Device	Equipment Number
*1	1721/1722	1
2	1729	1
3	1731-601	7 (A/Q channel)
*4	1738-853/854	3
5	1751	3
*6	1711/1712/1713	1
7	1742	F
8	Dummy	

4.2.3 SYSTEM UNIT ASSIGNMENTS

The standard system defines system units in LOG1A as follows:

System Unit	Lun
Input Comment	4
Output Comment	4
Binary Input	2
Binary Output	3
List	9
Library	8
Scratch	8

4.2.4 SYSTEM UNIT ASSIGNMENTS FOR SYSBFB, SYSBFC, SYSBFD

The following assignments coincide with the additional SYSBUF examples included on the COSY tape (SYSBFB, SYSBFC, SYSBFD).

Device	Lun
Core Allocator	1
Paper Tape Reader	2
Paper Tape Punch	3

Device	Lun
Teletypewriter	4
1729 Card Reader	5
Magnetic Tape Unit 0	6
Magnetic Tape Unit 1	7
Disk File	8
Printer	9
Dummy	10
1728-430 Reader/Punch	11
1729-2 Card Reader	12
Buffered Teletypewriter	13
Buffered Tape Punch	14
Buffer Printer	15
Buffered Card Punch	16

4.3 RELEASE TAPE FORMATS

4.3.1 MSOS 2.1 RELEASE TAPE FORMATS

MSOS 2.1 Initializer Tape

The Initializer MSOS 2.1 release tape contains two format records: record 1 is the Checksum Loader; record 2 is the System Initializer.

Record 1 Checksum Loader:

```
0045  P0000  6815              STA*   LOC
0046  P0001  0A20              ENA    $20
0047  P0002  E000              LDQ    =N$A1
      P0003  00A1
0048  P0004  03FE              OUT    -1
0049  P0005  5823              RTJ*   WORD1
0050  P0006  6811   BACK       STA*   IT
0051  P0007  5811   LOOP       RTJ*   WORD
0052  P0008  6C0D              STA*   (LOC)
0053  P0009  D80C              RAO*   LOC
0054  P000A  D80D              RAO*   IT
0055  P000B  C80C              LDA*   IT
0056  P000C  0101              SAZ    1
```

```
0057  P000D  18F9            JMP*   LOOP
0058  P000E  580A            RTJ*   WORD
0059  P000F  0000            SLS    0
0060  P0010  4807            STQ*   IT
0061  P0011  0181            SWS    1
0062  P0012  1C05            JMP*   (IT)
0063  P0013  5C04            RTJ*   (IT)
0064  P0014  0000            SLS    0
0065  P0015  0001            BZS    LOC,CHKSUM,IT
      P0016  0001
      P0017  0001
0066  P0018  0B00   WORD     NOP    0
0067  P0019  5809            RTJ*   GET
0068  P001A  0FC8   WORD2    ALS    8
0069  P001B  680D            STA*   TEMP
0070  P001C  5806            RTJ*   GET
0071  P001D  B80B            EOR*   TEMP
0072  P001E  0822            TRA    Q
0073  P001F  F8F6            ADQ*   CHKSUM
0074  P0020  48F5            STQ*   CHKSUM
0075  P0021  1CF6            JMP*   (WORD)
0076  P0022  0B00   GET      NOP    0
0077  P0023  E000            LDQ    =N$A0
      P0024  00A0
0078  P0025  0A00            ENA    0
0079  P0026  02FE            INP    -1
0080  P0027  1CFA            JMP*   (GET)
0081  P0028  0B00   WORD1    NOP    0
0082  P0029  0844            CLR    A
0083  P002A  68EB            STA*   CHKSUM
0084  P002B  C8FC            LDA*   WORD1
0085  P002C  68EB            STA*   WORD
0086  P002D  58F4   OVER     RTJ*   GET
0087  P002E  0111            SAN    1
0088  P002F  18FD            JMP*   OVER
0089  P0030  18E9            JMP*   WORD2
0090         0028 P TEMP     EQU    TEMP (WORD1)
0091                         END

I      00FF   BACK    0006P  LOOP    0007P  LOC    0015P  CHKSUM    0016P
IT     0017P  WORD    0018P  WORD2   001AP  GET    0022P  WORD1     0028P
OVER   002DP  TEMP    0028P
```

Record 2 System Initializer: Record 2 contains the System Initializer modules absolutized in absolute binary records by using the *P function of the LIBEDT routine. The System Initializer modules are listed in section II.4.3.2.

MSOS 2.1 Installation Tapes

The Installation MSOS 2.1 tapes contain relocatable programs and control cards.

Paper Tape 1

For non-buffered, non-re-entrant systems.

```
*S, ONE, 7FFF
*S, TWO, 7FFF
*S, THREE, 7FFF
*YM, LOADSD,1,JOBENT,2,JOBPRO,3,JPLOAD,4,JPST,5
*YM,JPCHGE,6,JBKILL,7,JPT13,8,MIPRO,9,LIBEDT,10
*YM,MOD1,11,MOD2,12,MOD3,13,MOD4,14,RESTOR,15
*YM,ODEBUG,16,RCOVER,17,BRKPT,18
*L        LOCORE
          LOCORE
          SYSBUF
          SCHEDU
          NDISP
          NCMPRQ
          NFNR
          ADEV
```

Paper Tape 2

```
    *M    LOADER
          LOAD
          BRANCH
          LIDRIV
          LCDRIV
          LMDRIV
          LLDRIV
          SCAN
          CHPU
          ADJOVF
          CONVRT
          TABSCH
          TABSTR
          LSTOUT
          LINK1
          LINK2
          COREXT
          DPRADD
          LOADER
          NAMPRO
          RBDBZS
```

```
*M      ENTEXT
        XFRPRO
        HEXPRO
        EOLPRO
        ADRPRO
```

Paper Tape 3

```
*L      DRCORE
        DRCORE
        ALCORE
        ALVOL
        OFVOL
        TRVEC
        PARAME
        COMMON
        NIPROC
        NEPROC
        NMONI
        RW
        MAKQ
        MINT
*M      JOBENT
        JOBENT
        T11
        T7
        T3
*M      JOBPRO
        JOBPRO
        PROTEC
        T5
*M      JPLOAD
        JPLOAD
*M      JPST
        JPST
*M      JPCHGE
        JPCHGE
        ASCHEX
```

Paper Tape 4

```
*M      JBKILL
        JBKILL
*M      JPT13
        JPT13
        T13
```

```
*M    MIPRO
      MIPRO
*M    LIBEDT
      LIBEDT
*M    UTILIB
      UTILIB
*M    PLINSN
      PLINSN
*M    FILE
      FILE
*M    GENLIB
      GENLIB
```

Paper Tape 5

```
*M    RESTORE DEVICE
      RESTOR
*M    ODEBUG
      ODEBUG
*M    RCOVER
      RCOVER
      OUTSEL
      DMPCOR
      MASDMP
*M    BRKPT
      BRKPT
      SIFT
      BIASCI
      RETJMP
      JUMPTO
      ENTER
      ENTCOR
      PRTREG
      TERMIN
      RESUME
      DMPCOR
      MASDMP
      SETBRP
```

Paper Tape 6

```
*L    DRIVERS
      DR1728
      CD1729
      PTREAD
      PUNCDR
      TELTYP
```

```
*L      TAPEDR
        FRWA
        FRWB
        RECOVT
        TAPE
        CARDRD
        PRINTR
        DISKWD
        SPACE
*S,TIMINT,7FFF
*S,SNAPE,7FFF
*S,PARITY,7FFF
*S,IPROC1,7FFF
*S,T30,7FFF
*S,T29,7FFF
*S,T28,7FFF
*S,T27,7FFF
*S,T26,7FFF
*S,T25,7FFF
*S,T24,7FFF
*S,T23,7FFF
*S,T22,7FFF
*S,T21,7FFF
*S,T20,7FFF
*S,T19,7FFF
*S,T18,7FFF
*S,T17,7FFF
*S,T16,7FFF
*S,T13,7FFF
*S,T11,7FFF
*S,T8,7FFF
*S,T7,7FFF
*S,T5,7FFF
*S,T3,7FFF
*S,JKIL,7FFF
*S,RWBA,7FFF
*S,RW609,7FFF
*S,DEBUG,7FFF
*S,DTIMER,7FFF
*T
*S,1,0,M
*S,2,1,M
*S,3,2,M
*S,4,3,M
*S,5,3,M
*S,6,3,M
*S,7,3,M
*S,8,3,M
*S,9,4,M
```

These are unpatched externals (entry points of programs not present in the normal system). To prevent an error printout, they are linked to 7FFF. If any of these modules are to be used, the *S statement associated with it should be deleted.

```
*S,10,2,M
*S,11,3,M
*S,12,3,M
*S,13,3,M
*S,14,3,M
*S,15,4,M
*S,16,5,M
*S,17,6,M
*S,18,0,M
*U
*ENDTAPE
```

COSY Source Tape MSOS 2.1

List of Deck names on MSOS 2.1 COSY tape in order of their occurrence.

```
LOCORE
SYSBUF
SCHEDU
NDISP
NCMPRQ
NFNR
ADEV
BUFFER
RDISP
SYSBFB
SYSBFC
SYSBFD
CONTRL
LIB
IDRIV
MTIDRV
PTIDRV
CDIDRV
MDRIV
MSDISK
MSDRUM
I2
I2DISK
I2DRUM
I1
ILOAD
CDRIV
LPRINT
TABLES
DRCORE
ALCORE
ALVOL
```

OFVOL
TRVEC
PARAME
COMMON
NIPROC
MIPROC
NEPROC
MEPROC
NMONI
MMONI
RW
MRW
MAKQ
MINT
TMINT
DTMER
SPACE
JOBENT
T11
T7
T3
JOBPRO
PROTEC
T5
JPLOAD
JPST
JPCHGE
ASCHEX
JBKILL
JPT13
T13
MIPRO
RESTOR
ODEBUG
LOAD
BRANCH
LIDRIV
LCDRIV
LMDRIV
LLDRIV
SCAN
CHPU
ADJOVF
CONVRT
TABSCH
TABSTR
LSTOUT
LINK1
LINK2

COREXT
DPRADD
LOADER
NAMPRO
RBDBZS
ENTEXT
XFRPRO
HEXPRO
EOLPRO
ADRPRO
RCOVER
OUTSEL
DMPCOR – For the recovery package
MASDMP – For the recovery package
BRKPTD
SIFT
BIASCI
RETJMP
JUMPTO
ENTER
ENTCOR
PRTREG
TERMIN
RESUME
DMPCOR – For the breakpoint package
MASDMP – For the breakpoint package
SETBRP
LIBEDT
UTILIB
PLINSN
FILE
GENLIB
PTREAD
PUNCDR
TELTYP
DRMDRZ
TAPDRB
RWBAB
FRWAB
FRWBB
RW609B
RECVTB
TAPEDR
FRWA
RW609
FRWB
RWBA
RECOVT
TAPE

```
    CARDRD
    DR1728
    CD1729
    PRINTR
    DISKWD
    DISK
```

This tape terminates with an EOF.

MSOS 2.1 List Tape I

```
    DECK  LOCORE
    DECK  SYSBUF
    DECK  SCHEDU
    DECK  NDISP
    DECK  NCMPRQ
    DECK  NFNR
    DECK  ADEV
    DECK  BUFFER
    DECK  RDISP
    DECK  SYSBFB
    DECK  SYSBFC
    DECK  SYSBFD
    DECK  CONTRL
    DECK  LIB
    DECK  IDRIV
    DECK  MTIDRV
    DECK  PTIDRV
    DECK  CDIDRV
    DECK  MDRIV
    DECK MSDISK
    DECK  MSDRUM
    DECK  I2
    DECK  12DISK
    DECK  12DRUM
    DECK  I1
    DECK  ILOAD
    DECK  CDRIV
    DECK  LPRINT
    DECK  TABLES
    DECK  DRCORE
    DECK  ALCORE
    DECK  ALVOL
    DECK  OFVOL
    DECK  TRVEC
    DECK  PARAME
    DECK  COMMON
    DECK  NIPROC
    DECK  MIPROC
    DECK  NEPROC
    DECK  MEPROC
    DECK  NMONI
    DECK  MMONI
    DECK  RW
    DECK  MRW
    DECK  MAKQ
    DECK  MINT
    DECK  TMINT
    DECK  DTMER
    DECK  SPACE
    DECK  JOBENT
    DECK  T11
    DECK  T7
    DECK  T3
    DECK  JOBPRO
    DECK  PROTEC
    DECK  T5
    DECK  JPLOAD
    DECK  JPST
    DECK  JPCHGE
    DECK  ASCHEX
    DECK  JBKILL
    DECK  JPT13
    DECK  T13
    DECK  MIPRO
    DECK  RESTOR
    DECK  ODEBUG
    EOF
```

MSOS 2.1 List Tape II

```
    DECK  LOAD
    DECK  BRANCH
    DECK  LIDRIV
    DECK  LCDRIV
    DECK  LMDRIV
    DECK  LLDRIV
    DECK  SCAN
    DECK  CHPU
    DECK  ADJOVE
    DECK  CONVRT
    DECK  TABSCH
    DECK  TABSTR
    DECK  LSTOUT
    DECK  LINK1
    DECK  LINK2
```

```
DECK  COREXT
DECK  DPRADD
DECK  LOADER
DECK  NAMPRO
DECK  RBDBZS
DECK  ENTEXT
DECK  XFRPRO
DECK  HEXPRO
DECK  EOLPRO
DECK  ADRPRO
DECK  RCOVER
DECK  OUTSEL
DECK  DMPCOR
DECK  MASDMP
DECK  BRKPTD
DECK  SIFT
DECK  BIASCI
DECK  RFTJMP
DECK  JUMPTO
DECK  ENTER
DECK  ENTCOR
DECK  PRTREG
DECK  TERMIN
DECK  RESUME
DECK  DMPCOR
DECK  MASDMP
DECK  SETBRP
DECK  LIBEDT
DECK  UTILIB
DECK  PLINSN
DECK  FILE
DECK  GENLIB
DECK  PTREAD
DECK  PUNCDR
DECK  TELTYP
DECK  DRMDRZ
DECK  TAPDRB
DECK  RWBAB
DECK  FRWAB
DECK  FRWBB
DECK  RW609B
DECK  RECVTB
DECK  TAPEDR
DECK  FRWA
DECK  RW609
DECK  FRWB
DECK  RWBA
DECK  RECOVT
DECK  TAPE
DECK  CARDRD
DECK  DR1728
DECK  CD1729
DECK PRINTR
DECK DISKWD
DECK DISK
EOF
```

4.3.2 MSOS 2.1 MODULE LIST

System Initializer Modules

CONTRL†	Control Module
LIB†	Library Generation Module
IDRIV†	Input Control Module (input device driver controller)
MDRIV†	Mass Storage Driver Control Module
CDRIV†	Comment Control Module (comment device driver)
ILOAD†	Resident Loader (relocatable binary loading module)
I1†	Pre-Resident Load Initialization
I2†	Initialization Controller Module 2; Post-Resident Load Initialization
MSDISK††	Pre-Resident Initialization 853/854 Disk Driver
MSDRUM††	Pre-Resident Initialization 1751 Drum Driver
12 DISK††	Post-Resident Initialization Disk Driver
12 DRUM††	Post-Resident Initialization Drum Driver
MTIDRV†††	601 Magnetic Tape Driver
PTIDRV†††	1721/1722 Paper Tape Reader Driver
LPRINT†††	1742 Line Printer Driver

Core Resident Modules

All of the following core resident modules are not included in a single system installation since this list includes all available core resident modules. The modules included in a particular system depend on the options desired.

† Required modules.

†† Normally either the disk or drum drivers are used, not both.

††† Optional according to the system configuration.

LOCORE[†]	16K LOCORE – predefined constants and interrupt slots
SYSBUF[†]	16K SYSBUF – system table
TABLES[†]	Minimum LOCORE and SYSBUF
BUFFER	Software buffering package
DRCORE	Core allocator driver
ALCORE	Core allocator
SCHEDU	Scheduler
NDISP	Normal dispatcher
RDISP	Dispatcher for use of the re-entrant FORTRAN
ALVOL	Volatile storage allocator
OFVOL	Volatile storage overflow error reporting
TRVEC	Transfer vectors
PARAME	Parameter conversion routines
COMMON	Common interrupt handler
NIPROC	Normal internal interrupt handler
MIPROC	Mini-internal interrupt handler
NEPROC	Normal external interrupt handler
MEPROC	Mini-external interrupt handler
NMONI	Normal monitor request processor
MMONI	Mini-monitor request processor
RW	Read/write request processor
MRW	Mini-read/write request processor
NCMPRQ	Normal complete request module
NFNR	Normal find next request module
MAKQ	Read/write Q generation module
ADEV	Alternate device handler
MINT	Manual interrupt handler
TIMINT	Timer driver
DTIMER	Diagnostic timer module
SPACE[†]	Core structuring module

[†] These modules will change with customization.

Loader Modules

LOAD	Initialization
BRANCH	Call tape
LIDRIV	Input driver
LCDRIV	Comment driver
LMDRIV	Mass storage driver
LLDRIV	List driver
SCAN	Unpack input
CHPU	Unpack input
ADJOVF	15-bit arithmetic
CONVRT	Binary to ASCII conversion
TABSCH	EXT, ENT, search
TABSTR	Loader table generation
LSTOUT	Message output
LINK1	Linkage operation 1
LINK2	Linkage operation 2
COREXT	ABS output to mass storage
DPRADD	Single precision to double precision ADD
LOADER	Loader control
NAMPRO	NAM processor
RBDBZS	RBD, BZS processor
EXTEXT	ENT, EXT processor
XFRPRO	XFR processor
HEXPRO	HEX processor
EOLPRO	EOL processor
ADRPRO	Address computation

Job Processor

Mass Memory Module JOBENT:

JOBENT	JOB processor entry module
T11	Core request processor

T7	Loader request processor
T3	Status request processor

Mass Memory Module JOBPRO:

JOBPRO	JOB processor control module
PROTEC	Protect processor
T5	Exit request processor

Mass Memory Module JPLOAD:

JPLOAD	Loader

Mass Memory Module JPST:

JPST	*B, *U, *SR, *V, *Z processor

Mass Memory Module JPCHGE:

JPCHGE	Logical unit change module
ASCHEX	ASCII conversion module

Mass Memory Module JBKILL:

JBKILL	JOB kill module

Mass Memory Module JPT13:

JPT13	JOB execution
T13	GTFILE request processor

Mass Memory Module LIBEDT:

LIBEDT	Control module

Mass Memory Module MOD1:

UTILIB	Utility functions

Mass Memory Module MOD2:

PLINSN	Program library construction

Mass Memory Module MOD3:

FILE	File generation

Mass Memory Module MOD4:

GENLIB	Transfer functions

Miscellaneous Mass Memory Modules

Mass Memory Module MIPRO:

MIPRO	Manual interrupt processor

Mass Memory Module RESTOR:

RESTOR	Restores logical units

Mass Memory Module ODEBUG:

ODEBUG	On-line debug module

Mass Memory Module RCOVER:

RCOVER	Control module
OUTSEL	Output unit select module
DMPCOR	Core dump module
MASDMP	Mass storage dump module

Mass Memory Module BRKPT:

BRKPTD	Control module
SIFT	Statement analyzer
BIASCI	Binary ASCII conversion
RETJMP	*R statement processor
JUMPTO	*J statement processor
ENTER	*A, *Q, *J statement processor
ENTCOR	*E statement processor
PRTREG	*P statement processor
TERMIN	*T statement processor
RESUME	*C statement processor
DMPCOR	Core dump module
MASDMP	Mass memory dump module
SETBRP	*S statement processor

Available Drivers

PTREAD	1723/1724 paper tape reader
PUNCDR	1721/1722 paper tape punch
TELETYP	1711/1712/1713 teletype
DRMDRZ	1751 drum
TAPDRB	1731/1732-1706-601-608 buffered magnetic tape control
RWBAB	1731/1732-1706-601-608 buffered non-format read/write
FRWAB	1731/1732-1706-601-608 buffered format ASCII read/write
FRWBB	1731/1732-1706-609 buffered format binary read/write
RW609B	1731/1732-601/608/609 buffered ASCII binary read/write
RECVTB	1731-1706 recovery
TAPEDR	1731/1732-601/608 magnetic tape control
FRWA	1731/1732-601/608 format ASCII read/write
RW609	1731/1732-601/608/609 ASCII binary read/write
FRWB	1731/1732-601/608 format binary read/write
RWBA	1731/1732-601/608 non-format read/write
RECOVT	1731/1732 recovery
TAPE	1731/1732 tape motion control
CARDRD	1729 card reader
DR1728	1728-430 card reader driver
CD1729	1729-2 card reader driver
PRINTR	1742 printer
DISKWD	1738-853/854 disk word driver
DISK	1738-853/854 disk driver
MASDRV	Mass memory control program for drivers including 1713 teletypewriter
S13001	1713 teletypewriter keyboard module
S13002	1713 teletypewriter reader module
S13003	1713 teletypewriter punch module

4.3.3 MACRO ASSEMBLER 2.0 RELEASE TAPE FORMATS

Tape 1 and Tape 4

Paper tape 1 and magnetic tape 4 are the same except for control statements which assign logical units.

```
*K, I6, P8 (I2 for the paper tape)
*L, ASSEM
    Relocatable deck for ASSEM
*P, F
    Relocatable decks for PASS 1
*T
*K, I8
*N, PASS1, , , B
*K, I6
    ...
    ...Same for PASSES 2-4
    ...
*NMACSKL, , , B
    Absolute MACSKL for library macros
*N, MACROS, , , B
    Absolute MACROS for library macros
*U
```

Tape 2

Tape 2 contains LIBRARY macro preparation programs and relocatable binary of LIBMAC, LIBMC2, and LIBMC3.

Tape 3

Tape 3 is the Macro Assembler source tape and contains the source of the LIBRARY macros in the following order:

FREAD

FWRITE

Q8A

Q8B

STATUS

READ

WRITE

INDIR

EXIT

CORE

LOADER

SCHDLE

TIMER

GTFILE

SPACE

RELEAS

Tape 5

Tape 5 is an optional magnetic tape containing in COSY format the sources for ASSEM, passes 1 through 4, and LIBMAC.

COSY Deck ID	Program	COSY Deck Name
AS	ASSEM	ASSEM
OO	PASS1	PASS1
OW	PA1PR2	PA1PR2
WO	PASS2	PASS2
WW	PA2PR2	PA2PR2
TO	PASS3	PASS3
TW	PA3PR2	PA3PR2
TT	PA3PR3	PA3PR3
FO	PASS4	PASS4
LB1	LIBMAC	LIBMAC
LB2	LIBMC2	LIBMC2
LB3	LIBMC3	LIBMC3

Tape 6

Tape 6 is a Hollerith listing of COSY programs.

4.3.4 MASS STORAGE FORTRAN 2.0A RELEASE TAPE FORMATS

FORTRAN 2.0A Installation Tape Format

The installation tape has the following format for magnetic tape. For paper tape, *K, I6, P8 is replaced by *K, I2, P8 and there is an *U at the end of each physical tape.

```
*K,I6,P8
*P
            FTN
            GOA
            CNVT
            CONV
            DIAG
            EXP9
            FLOAT
            GETSYM
            GPUT
            IOPRBA
            PACK
            Q8PRMS
            STORE
            SYMBOL
            LOCLA1
            DUMYA1
            ENDDO
            GETC
            GETF
            GNST
            IGETCF
            OPTION
            OUTENT
            PHASEA
            PLABEL
            Q8QBDS
            RDLABL
            STCHAR
            TYPE
            ENDLOC
*T
*K,I8
*N,FORTA1,,,B
*K,I6,P8
*P
            FTN
            GOA
            CNVT
            CONV
```

```
          DIAG
          EXP9
          FLOAT
          GETSYM
          GPUT
          IOPRBA
          PACK
          Q8PRMS
          STORE
          SYMBOL
          LOCLA2
          DUMYA2
          ARITH
          COMNPR
          DIMPR
          GETC
          GETF
          SUBSCR
          TYPEPR
          ENDLOC
*T
*K,I8
*N,FORTA2,,,B
*K,I6,P8
*P
          FTN
          GOA
          CNVT
          CONV
          DIAG
          EXP9
          FLOAT
          GETSYM
          GPUT
          IOPRBA
          PACK
          Q8PRMS
          STORE
          SYMBOL
          LOCLA3
          DUMYA3
          BYEQPR
          CHECKF
          CONSUB
          DATAPR
          FGETC
          FORK
          GETC
          GETF
          STCHAR
          TREE
          ENDLOC
*T
*K, I8
*N, FORTA3, , , B
*K, I6, P8
*P
          FTN
          GOA
          CNVT
          CONV
          DIAG
          EXP9
          FLOAT
          GETSYM
          GPUT
          IOPRBA
          PACK
          Q8PRMS
          STORE
          SYMBOL
          LOCLA4
          DUMYA4
          ARAYSZ
          ASGNPR
          BDOPR
          CFIVOC
          CKIVC
          CKNAME
          CPLOOP
          ENDDO
          GETC
          GETF
          IOSPR
          OUTENT
          RDLABL
          STCHAR
          ENDLOC
*T
*K, I8
*N, FORTA4, , , B
*K, I6, P8
*P
          FTN
          GOA
          CNVT
          CONV
          DIAG
          EXP9
          FLOAT
          GETSYM
          GPUT
          IOPRBA
          PACK
          Q8PRMS
          STORE
          SYMBOL
          LOCLA5
          DUMYA5
          ARITH
          GETC
          GETF
          SUBSCR
          ENDLOC
*T
*K,I8
*N, FORTA5,,,B
*K,I6,P8
*P
          FTN
          GOA
          CNVT
          CONV
          DIAG
          EXP9
          FLOAT
          GETSYM
          GPUT
          IOPRBA
          PACK
          Q8PRMS
          STORE
          SYMBOL
          LOCLA6
          DUMYA6
          CFIVOC
          CKIVC
          ERBPR
          GETC
          GETF
          MODMXR
          RDLABL
          SUBPPR
          TREE
          ENDLOC
```

```
*T
*K, I8
*N, FORTA6, , , B
*K, I6, P8
*P
          FTN
          GOA
          CNVT
          CONV
          DIAG
          EXP9
          FLOAT
          GETSYM
          GPUT
          IOPRBA
          PACK
          Q8PRMS
          STORE
          SYMBOL
          LOCLA7
          DUMYA7
          ASEMPR
          EXRLPR
          GETC
          GETF
          IGETCF
          PEQVS
          PRNTNM
          PUNT
          RDLABL
          SYMSCN
          ENDLOC
*T
*K,I8
*N, FORTA7, , , B
*K, I6, P8
*P
          FTN
          GOB
          CNVT
          DUMMY
          FCMSTK
          GETSYM
          IOPRBB
          KCPART
          KOUTPT
          KPCSTK
          KPC3PR
          KSYMGN
```

```
        LABKPC
        LABLER
        PUNT
        Q8PRMS
        STORE
        SYMBOL
        TSALOC
        LOCLB1
        DUMYB1
        ARAYSZ
        ASSEM
        BANANA
        BGINDO
        END
        ENTCOD
        HELEN
        INXRST
        NOPROC
        PHASEB
        READIR
        SUBFUN
        SYMSCN
        ENDLOC
*T
*K,I8
*N,FORTB1,,,B
*K,I6,P8
*P
        FTN
        GOB
        CNVT
        DUMMY
        FCMSTK
        GETSYM
        IOPRBB
        KCPART
        KOUTPT
        KPCSTK
        KPC3PR
        KSYMGN
        LABKPC
        LABLER
        PUNT
        Q8PRMS
        STORE
        SYMBOL
        TSALOC
        LOCLB2
```

```
          ACP
          AFIDL
          ASUPER
          CGOTO
          FINK
          INTRAM
          PARTSB
          SUBPR1
          SUBPR2
          SUBPR3
          ENDLOC
*T
*K,I8
*N,FORTB2,,,B
*K,I6,P8
*P
          FTN
          GOB
          CNVT
          DUMMY
          FCMSTK
          GETSYM
          IOPRBB
          KCPART
          KOUTPT
          KPCSTK
          KPC3PR
          KSYMGN
          LABKPC
          LABLER
          PUNT
          Q8PRMS
          STORE
          SYMBOL
          TSALOC
          LOCLB3
          ACP
          ARITHR
          ASUPER
          FINK
          INTRAM
          PARTSB
          SUBPR1
          SUBPR2
          SUBPR3
          ENDLOC
```

```
*T
*K,I8
*N,FORTB3,,,B
*K,I6,P8
*P
        FTN
        GOC
        BKDWN
        BLDUP
        BSS
        CHKWD
        CHOP
        CL12
        CON
        COUNT
        DATAST
        GETSYM
        INOUT
        IXOPT
        PHASEC
        LABEL
        LABIN
        QXLD
        REED
        SKIP
        SYMSCN
        IOPRBC
        Q8PRMS
        ENDLOC
*T
*K,I8
*N,FORTC1,,,B
*K,I6,P8
*P
        FTN
        GOOD
        INDEX
        IOPRBD
        NPUNCH
        Q8PRMS
        PHASE6
        LOCLD1
        DUMYD1
        AMT
        AMOUT
        ADMAX
        BKDWN
        COUNT
        LABOUT
```

```
           NP2OUT
           RBDX
           RBPK
           TABDEC
           UNPUNC
           GETSYM
           SYMSCN
           ENDLOC
*T
*K,I8
*N,FORTD1,,,B
*K,I6,P8
*P
           FTN
           GOOD
           INDEX
           IOPRBD
           NPUNCH
           Q8PRMS
           PHASE6
           LOCLD2
           DUMYD2
           AMT
           GETSYM
           IACON
           IHCON
           NWRITE
           PACK
           SYMSCN
           BEGINO
           FINISH
           ENDLOC
*T
*K,I8
*N,FORTD2,,,B
*K,I6,P8
*P
           FTN
           GOE
           INDEX
           IOPRBD
           NPUNCH
           Q8PRMS
           PHASE6
           LOCLD1
           DUMYD1
           AMT
           AMOUT
           ADMAX
```

```
          BKDWN
          COUNT
          LABOUT
          NP2OUT
          RBDX
          RBPK
          TABDEC
          UNPUNC
          CONV
          GETSYM
          IACON
          IHCON
          NWRITE
          PACK
          SETPRT
          SYMSCN
          ENDLOC
*T
*K,I8
*N,FORTE1,,,B
*K,I6,P8
*P
          FTN
          GOE
          INDEX
          IOPRBD
          NPUNCH
          Q8PRMS
          PHASE6
          LOCLD2
          DUMYD2
          AMT
          CONV
          GETSYM
          IACON
          IHCON
          NWRITE
          PACK
          SETPRT
          SYMSCN
          BEGINO
          FINISH
          ENDLOC
*T
*K,I8
*N,FORTE2,,,B
*K,I6,P8
*L,FTN
           FTN
```

```
*L,Q8IFRM
          Q8IFRM
*L,Q8FS
          Q8FS
*L,Q8TRAN
          Q8TRAN
*L,FLOT
          FLOAT
*L,Q8QINI
          Q8QINI
*L,Q8QEND
          Q8QEND
*L,Q8CMP1
          Q8CMP
*L,Q8RWBU
          Q8RWBU
*L,Q8ERRM
          Q8ERRM
*L,Q8DFNF
          Q8DFIO
*L, Q8QX
          Q8QX
*L,Q8QUNI
          Q8QUNI
*L, Q8FGET
          Q8FGET
*L,Q8MAGT
          Q8MAGT
*L,Q8GBCK
          TAPCON
*L,IOCK
          IOCK
*L,Q8PSE
          PSSTOP
*L,Q8PAND
          Q8PAND
*L,Q8EXP9
          Q8EXP9
*L,Q8EXP1
          Q8EXP1
*L,Q8AB
          Q8AB
*L,SIGN
          SIGN
*L,EXP
          EXPPRG
*L,SQRT
          SQRTF
```

```
*L,ALOG
        LNUPRG
*L,TANH
        TANH
*L,SIN
        SINCOS
*L,ATAN
        ARCTPG
*L,QSAVE
        Q8EXPN
*L,IFALT
        IFALT
*L,Q8FX
        FXFL
*L,Q8PREP
        Q8PRMS
*U
```

FORTRAN 2.0A COSY Source Magnetic Tape

The FORTRAN 2.0A source tape is in COSY (compressed) format. The programs are arranged in the following order:

1. Phase A programs written in FORTRAN.
2. Phase B programs written in FORTRAN.
3. Phases C, D, E programs written in FORTRAN.
4. Phase A, B, C, D, E programs written in assembly language.
5. Object library programs written in FORTRAN.
6. Object library programs written in assembly language.

To assemble or to compile a program, convert from COSY format into Hollerith format and then work with the Hollerith tape. Following is a list of sequence numbers and COSY deck names of the FORTRAN routine names.

Phase A programs written in FORTRAN:

COSY Deck Identifier	Program	COSY Deck Name	Phases
A1	CNVT	CNVT	A1, A2, A3, A4, A5, A6, A7, B1, B2, B3
A2	GPUT	GPUT	A1, A2, A3, A4, A5, A6, A7
A3	SYMBOL	SYMBL1	A1, A2, A3, A4, A5, A6, A7
A4	†GETF	GETF1	A1, A2, A3, A4, A6, A7

†Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program	COSY Deck Name	Phases
A5	GNST	GNST	A1
A6	OUTENT	OUTENT	A1, A4
A7	PHASEA	PHASEA	A1
A8	PLABEL	PLABEL	A1
A9	Q8QBDS	Q8QBDS	A1
A10	RDLABL	RDLABL	A1, A4, A6, A7
A11	STCHAR	STCHAR	A1, A3, A4
A12	TYPE	TYPE	A1
A13	†ARITH	ARITH1	A2
A14	COMNPR	COMNPR	A2
A15	DIMPR	DIMPR	A2
A16	SUBSCR	SUBSCR	A2, A5
A17	TYPEPR	TYPEPR	A2
A18	BYEQPR	BYEQPR	A3
A19	CHECKF	CHECKF	A3
A20	FGETC	FGETC	A3
A21	FORK	FORK	A3
A22	†ARITH	ARITH2	A5
A23	†GETF	GETF2	A5
A24	SUBPPR	SUBPPR	A6
A25	EXRLPR	EXRLPR	A7
A26	PEQVS	PEQVS	A7
A27	PRNTNM	PRNTNM	A7
A28	*PUNT	PUNT1	A7
A29	*SYMSCN	SMSCN4	A7, B1
A30	ENDDO	ENDDO	A1, A4
A31	CONSUB	CONSUB	A3
A32	DATAPR	DATAPR	A3
A33	ASGNPR	ASGNPR	A4
A34	BDOPR	BDOPR	A4

†Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program	COSY Deck Name	Phases
A35	CFIVOC	CFIVOC	A4, A6
A36	CKIVC	CKIVC	A4, A6
A37	CKNAME	CKNAME	A4
A38	IOSPR	IOSPR	A4
A39	ERBPR	ERBPR	A6
A40	MODMXR	MODMXR	A6
A41	ASEMPR	ASEMPR	A7
A42	TREE	TREE	A3, A6
A43	ARAYSZ	ARAYSZ	A4, B1
A44	CPLOOP	CPLOOP	A4

Phase B programs written in FORTRAN:

COSY Deck Identifier	Program	COSY Deck Name	Phases
A50	DUMMY	DUMMY	B1, B2, B3
A51	FCMSTK	FCMSTK	B1, B2, B3
A52	KCPART	KCPART	B1, B2, B3
A53	KOUTPT	KOUTPT	B1, B2, B3
A54	KPCSTK	KPCSTK	B1, B2, B3
A55	KPC3PR	KPC3PR	B1, B2, B3
A56	KSYMGN	KSYMGN	B1, B2, B3
A57	LABKPC	LABKPC	B1, B2, B3
A58	LABLER	LABLER	B1, B2, B3
A59	†PUNT	PUNT2	B1, B2, B3
A60	†SYMBOL	SYMBL2	B1, B2, B3
A61	TSALOC	TSALOC	B1, B2, B3
A62	ASSEM	ASSEM	B1
A63	BANANA	BANANA	B1
A64	BGINDO	BGINDO	B1
A65	END	END	B1

†Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program	COSY Deck Name	Phases
A66	ENTCOD	ENTCOD	B1
A67	HELEN	HELEN	B1
A68	INXRST	INXRST	B1
A69	NOPROC	NOPROC	B1
A70	PHASEB	PHASEB	B1
A71	READIR	READIR	B1
A72	SUBFUN	SUBFUN	B1
A73	ACP	ACP	B2, B3
A74	AFIDL	AFIDL	B2
A75	ASUPER	ASUPER	B2, B3
A76	CGOTO	CGOTO	B2
A77	FINK	FINK	B2, B3
A78	INTRAM	INTRAM	B2, B3
A79	PARTSB	PARTSB	B2, B3
A80	SUBPR1	SUBPR1	B2, B3
A81	SUBPR2	SUBPR2	B2, B3
A82	SUBPR3	SUBPR3	B2, B3
A83	ARITHR	ARITHR	B3

Phases C, D, and E programs written in FORTRAN:

COSY Deck Identifier	Program	COSY Deck Name	Phases
B01	†BKDWN	BKDWN1	C
B02	BLDUP	BLDUP	C
B03	BSS	BSS	C
B04	CHKWD	CHKWD	C
B05	CHOP	CHOP	C
B06	CL12	CL12	C
B07	CON	CON	C
B08	†COUNT	COUNT1	C

†Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program	COSY Deck Name	Phases
B09	DATAST	DATAST	C
B10	†GETSYM	GTSYM1	C
B11	INOUT	INOUT	C
B12	IXOPT	IXOPT	C
B13	PHASEC	PHASEC	C
B14	LABEL	LABEL	C
B15	LABIN	LABIN	C
B16	QXLD	QXLD	C
B17	REED	REED	C
B18	SKIP	SKIP	C
B19	†SYMSCN	SMSCN1	C
B20	†INDEX	INDEX1	D1, D2
B21	†NPUNCH	NPNCH1	D1, D2
B22	†PHASE6	PHSE61	D1, D2
B23	†AMT	AMT1	D1, E1
B24	†AMOUT	AMOUT1	D1
B25	†ADMAX	ADMAX1	D1
B26	†BKDWN	BKDWN2	D1
B27	†COUNT	COUNT2	D1
B28	†LABOUT	LBOUT1	D1
B29	†NP2OUT	NP2OT1	D1
B30	†RBDX	RBDX1	D1
B31	†RBPK	RBPK1	D1
B32	†TABDEC	TBDEC1	D1
B33	†UNPUNC	UNPNC1	D1
B34	†GETSYM	GTSYM2	D1
B35	†SYMSCN	SMSCN2	D1
B36	†AMT	AMT2	D2, E2
B37	†GETSYM	GTSYM3	D2
B38	†IACON	IACON1	D2

†Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program	COSY Deck Name	Phases
B39	†IHCON	IHCON1	D2
B40	†NWRITE	NWRTE1	D2
B41	†SYMSCN	SMSCN3	D2, E1, E2
B42	†BEGINO	BEGNO1	D2
B43	†FINISH	FNISH1	D2
B44	†INDEX	INDEX2	E1, E2
B45	†NPUNCH	NPNCH2	E1, E2
B46	†PHASE6	PHSE62	E1, E2
B47	†AMOUT	AMOUT2	E1
B48	†ADMAX	ADMAX2	E1
B49	†BKDWN	BKDWN3	E1
B50	†COUNT	COUNT3	E1
B51	†LABOUT	LBOUT2	E1
B52	†NP2OUT	NP2OT2	E1
B53	†RBDX	RBDX2	E1
B54	†RBPK	RBPK2	E1
B55	†TABDEC	TBDEC2	E1
B56	†UNPUNC	UNPNC2	E1
B57	†GETSYM	GTSYM4	E1, E2
B58	†IACON	IACON2	E1, E2
B59	†IHCON	IHCON2	E1, E2
B60	†NWRITE	NWRTE2	E1, E2
B61	SETPRT	SETPRT	E1, E2
B62	†BEGINO	BEGNO2	E2
B63	†FINISH	FNISH2	E2

Compiler programs written in assembly language:

COSY Deck Identifier	Program	COSY Deck Name	Phases
C01	FTN	FTN4	A1, A2, A3, A4, A5, A6, A7, B1, B2, B3, C, D1, D2, E1, E2

†Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program	COSY Deck Name	Phases
C02	GOA	GOA	A1, A2, A3, A4, A5, A6, A7
C03	†CONV	CONV1	A1, A2, A3, A4, A5, A6, A7
C04	DIAG	DIAG	A1, A2, A3, A4, A5, A6, A7
C05	EXP9	EXP9	A1, A2, A3, A4, A5, A6, A7
C06	†FLOAT	FLOAT1	A1, A2, A3, A4, A5, A6, A7
C07	†GETSYM	GTSYM5	A1, A2, A3, A4, A5, A6, A7, B1, B2, B3
C08	IOPRBA	IOPRBA	A1, A2, A3, A4, A5, A6, A7
C09	PACK	PACK	A1, A2, A3, A4, A5, A6, A7, D2, E1, E2
C10	Q8PRMS	Q8PRMS	A1, A2, A3, A4, A5, A6, A7, B1, B2, B3, C, D1, D2, E1, E2, Object Library
C11	STORE	STORE	A1, A2, A3, A4, A5, A6, A7, B1, B2, B3
C12	LOCLA1	LOCLA1	A1
C13	DUMYA1	DUMYA1	A1
C14	†GETC	GETC1	A1, A2, A3, A4, A6, A7
C15	IGETCF	IGETCF	A1, A7
C16	OPTIONS	OPTION	A1
C17	ENDLOC	ENDLOC	A1, A2, A3, A4, A5, A6, A7, B1, B2, B3, C, D1, D2, E1, E2
C18	LOCLA2	LOCLA2	A2
C19	DUMYA2	DUMYA2	A2
C20	LOCLA3	LOCLA3	A3
C21	DUMYA3	DUMYA3	A3
C22	LOCLA4	LOCLA4	A4
C23	DUMYA4	DUMYA4	A4
C24	LOCLA5	LOCLA5	A5
C25	DUMYA5	DUMYA5	A5
C26	†GETC	GETC2	A5
C27	LOCLA6	LOCLA6	A6
C28	DUMYA6	DUMYA6	A6
C29	LOCLA7	LOCLA7	A7
C30	DUMYA7	DUMYA7	A7

†Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program	COSY Deck Name	Phases
C31	GOB	GOB	B1, B2, B3
C32	IOPRBB	IOPRBB	B1, B2, B3
C33	LOCLB1	LOCLB1	B1
C34	DUMYB1	DUMYB1	B1
C35	LOCLB2	LOCLB2	B2
C36	LOCLB3	LOCLB3	B3
C37	GOC	GOC	C
C38	IOPRBC	IOPRBC	C
C39	GOOD	GOOD	D1, D2
C40	IOPRBD	IOPRBD	D1, D2, E1, E2
C41	LOCLD1	LOCLD1	D1, E1
C42	DUMYD1	DUMYD1	D1, E1
C43	LOCLD2	LOCLD2	D2, E2
C44	DUMYD2	DUMYD2	D2, E2
C45	GOE	GOE	E1, E2
C46	†CONV	CONV2	E1, E2

Object library programs written in FORTRAN:

COSY Deck Identifier	Program	COSY Deck Name
Q01	Q8IFRM	Q8IFRM
Q02	Q8FS	Q8FS
Q03	Q8TRAN	Q8TRAN

Object library programs written in assembly language:

COSY Deck Identifier	Program	COSY Deck Name
Q04	†FLOAT	FLOAT2
Q05	Q8QINI	Q8QINI

† Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program	COSY Deck Name
Q06	Q8QEND	Q8QEND
Q07	Q8CMP	Q8CMP
Q08	Q8RWBU	Q8RWBU
Q09	Q8ERRM	Q8ERRM
Q10	Q8DFIO	Q8DFIO
Q11	Q8QX	Q8QX
Q12	Q8QUNI	Q8QUNI
Q13	Q8FGET	Q8FGET
Q14	Q8MAGT	Q8MAGT
Q15	TAPCON	TAPCON
Q16	IOCK	IOCK
Q17	PSSTOP	PSSTOP
Q18	Q8PAND	Q8PAND
Q19	Q8EXP9	Q8EXP9
Q20	Q8EXP1	Q8EXP1
Q21	Q8AB	Q8AB
Q22	SIGN	SIGN
Q23	EXPPRG	EXPPRG
Q24	SQRTF	SQRTF
Q25	LNUPRG	LNUPRG
Q26	TANH	TANH
Q27	SINCOS	SINCOS
Q28	ARCTPG	ARCTPG
Q29	Q8EXPN	Q8EXPN
Q30	IFALT	IFALT
Q31	FXFL	FXFL

4.3.5 MASS STORAGE FORTRAN 2.0A COMPILER PROGRAM ORDER

The compiler consists of four passes over the source code or its equivalent, accomplished in four phases called A, B, C, and D/E. (The fourth pass is performed by either Phase D or Phase E depending upon whether the user wants to use an assembly language listing output.) Each phase consists of a root which is core-resident throughout the phase, and zero or more local subroutine groups which share the same core area and are read from disc as needed. Phase A reads the source input, converts it to statements expressed in an internal code, and assigns a statement number to the statement.

Phase B reads the output of Phase A and generates pseudo code from it. (Pseudo code is similar to assembler input except that the index to be used in an indexed instruction and the addressing mode are not specified.)

Phase C and D/E are a two-pass assembler. The output from Phase B is read. Index registers are optimally assigned. One word relative addressing is maximized. Relocatable binary output and an assembly listing are produced.

Files	Root	Local
FORTA1	A	1
FORTA2	A	2
FORTA3	A	3
FORTA4	A	4
FORTA5	A	5
FORTA6	A	6
FORTA7	A	7
FORTB1	B	1
FORTB2	B	2
FORTB3	B	3
FORTC1	C	---
FORTD1	D	1
FORTD2	D	2
FORTE1	E	1

Phase A Programs

Root Programs:

FTN
GOA
CNVT
CONV

DIAG
EXP9
FLOAT
GETSYM
GPUT
IOPRBA
PACK
Q8PRMS
STORE
SYMBOL

Local 1 Programs:

LOCLA1
DUMYA1
ENDDO
GETC
GETF
GNST
IGETCF
OPTION
OUTENT
PHASEA
PLABEL
Q8QBDS
RDLABL
STCHAR
TYPE
ENDLOC

Local 2 Programs:

LOCLA2
DUMYA2
ARITH
COMNPR
DIMPR
GETC
GETF
SUBSCR
TYPEPR
ENDLOC

Local 3 Programs:

LOCLA3
DUMYA3
BYEQPR
CHECKF

CONSUB
DATAPR
FGETC
FORK
GETC
GETF
STCHAR
TREE
ENDLOC

Local 4 Programs:

LOCLA4
DUMYA4
ARAYSZ
ASGNPR
BDOPR
CFIVOC
CKIVC
CKNAME
CPLOOP
ENDDO
GETC
GETF
IOSPR
OUTENT
RDLABL
STCHAR
ENDLOC

Local 5 Programs:

LOCLA5
DUMYA5
ARITH
GETC
GETF
SUBSCR
ENDLOC

Local 6 Programs:

LOCLA6
DUMYA6
CFIVOC
CKIVC
ERBPR
GETC
GETF

MODMXR
RDLABL
SUBPPR
TREE
ENDLOC

Local 7 Programs:

LOCLA7
DUMYA7
ASEMPR
EXRLPR
GETC
GETF
IGETCF
PEQVS
PRNTNM
PUNT
RDLABL
SYMSCN
ENDLOC

Pass B Programs

Root Programs:

FTN
GOB
CNVT
DUMMY
FCMSTK
GETSYM
IOPRBB
KCPART
KOUTPT
KPCSTK
KPC3PR
KSYMGN
LABKPC
LABLER
PUNT
Q8PRMS
STORE
SYMBOL
TSALOC

Local 1 Programs:

LOCLB1
DUMYB1
ARAYSZ
ASSEM
BANANA
BGINDO
END
ENTCOD
HELEN
INXRST
NOPROC
PHASEB
READIR
SUBFUN
SYMSCN
ENDLOC

Local 2 Programs:

LOCLB2
ACP
AFIDL
ASUPER
CGOTO
FINK
INTRAM
PARTSB
SUBPR1
SUBPR2
SUBPR3
ENDLOC

Local 3 Programs:

LOCLB3
ACP
ARITHR
ASUPER
FINK
INTRAM
PARTSB
SUBPR1
SUBPR2
SUBPR3
ENDLOC

Pass C Programs

FTN
GOC
BKDWN
BLDUP
BSS
CHKWD
CHOP
CL12
CON
COUNT
DATAST
GETSYM
INOUT
IXOPT
PHASEC
LABEL
LABIN
QXLD
REED
SKIP
SYMSCN
IOPRBC
Q8PRMS
ENDLOC

Pass D Programs

Root Programs:

FTN
GOOD
INDEX
IOPRBD
NPUNCH
Q8PRMS
PHASE6

Local 1 Programs:

LOCLD1
DUMYD1
AMT
AMOUT
ADMAX
BKDWN
COUNT
LABOUT

NP2OUT
RBDX
RBPK
TABDEC
UNPUNC
GETSYM
SYMSCN
ENDLOC

Local 2 Programs:

LOCLD2
DUMYD2
AMT
GETSYM
IACON
IHCON
NWRITE
PACK
SYMSCN
BEGINO
FINISH
ENDLOC

Pass E Programs

Root Programs:

FTN
GOE
INDEX
IOPRBD
NPUNCH
Q8PRMS
PHASE6

Local 1 Programs:

LOCLD1
DUMYD1
AMT
AMOUT
ADMAX
BKDWN
COUNT
LABOUT
NP2OUT
RBDX
RBPK

TABDEC
UNPUNC
CONV
GETSYM
IACON
IHCON
NWRITE
PACK
SETPRT
SYMSCN
ENDLOC

Local 2 Programs:

LOCLD2
DUMYD2
AMT
CONV
GETSYM
IACON
IHCON
NWRITE
PACK
SETPRT
SYMSCN
BEGINO
FINISH
ENDLOC

4.3.6 MASS STORAGE FORTRAN 2.0A COMPILER PROGRAM LENGTHS, COMMON LENGTHS AND EXTERNALS

FORTRAN 2.0A Compiler Programs: Phase A

Program	Common		Program	
Name	Labeled	Blank	Length	Externals
CNVT	1649	3	62	------
GPUT	1649	1236	41	------
SYMBOL	1649	3	162	GETSYM WRITE SKIPIT
GETF	1649	1236	724	GETC GPUT DIAG EXP9 CNVT SYMBOL

Program Name	Common Labeled	Common Blank	Program Length	Externals
GNST	1649	1236	446	CONV
				PACK
				WRITE
				IGETCF
				READ
				STCHAR
				EXIT
				DIAG
				SKIPIT
QUTENT	1649	1236	52	Q8PRUP
				Q8PREP
				WRITE
PHASEA	1649	1236	1259	WRITE
				GNST
				PLABEL
				TYPE
				DIAG
				OUTENT
				PEQVS
				DIMPR
				COMNPR
				TYPEPR
				SUBPPR
				GETF
				STORE
				BYEQPR
				EXRLPR
				GETC
				DATAPR
				CHECKF
				ARITH
				GETSYM
				SYMBOL
				ASGNPR
				RDLABL
				ENDDO
				CPLOOP
				SKIPIT
				ERBPR
				IOSPR
				BDOPR
				STCHAR
				ASEMPR

Program Name	Common		Program Length	Externals
	Labeled	Blank		
PLABEL	1649	1236	86	RDLABL STORE DIAG
Q8QBDS	1649	----	0	------
RDLABL	1649	1236	129	GETC CNVT SYMBOL
STCHAR	----	----	50	------
TYPE	1649	1236	510	Q8PRUP Q8PREP GETC GETF
ARITH	1649	1236	1637	GETF DIAG PUNT TREE GETSYM SUBSCR STORE GETC SYMBOL
COMNPR	1649	1236	150	GETF DIMPR DIAG
DIMPR	1649	1236	391	GETF DIAG STORE GETSYM
SUBSCR	1649	1236	701	GETF DIAG STORE PUNT GETSYM SYMBOL
TYPEPR	1649	1236	23	DIMPR
BYEQPR	1649	1236	499	GETF DIAG STORE PUNT GETSYM

Program Name	Common Labeled	 Blank	Program Length	 Externals
CHECKF	1649	1236	160	FORK DIAG FGETC
FGETC	1649	1236	31	Q8PKUP Q8PREP GETC STCHAR
FORK	----	----	370	FGETC
ARITH	1649	1236	1630	GETF DIAG PUNT TREE GETSYM SUBSCR STORE GETC SYMBOL
GETF	1649	1236	724	GETC GPUT DIAG EXP9 CNVT SYMBOL
SUBPPR	1649	1236	192	GETF DIAG STORE
EXPLPR	1649	1236	94	GETF DIAG STORE
PEQVS	1649	1236	991	SYMSCN GETSYM PRNTNM DIAG
PRNTNM	1649	----	141	Q8PKUP Q8PREP GETSYM WRITE
PUNT	----	1236	56	DIAG READ IGETCF SKIPIT

Program	Common		Program	
Name	Labeled	Blank	Length	Externals
SYMSCN	1553	----	28	GETSYM
ENDDO	1649	1236	261	GETSYM OUTENT DIAG
CONSUB	1649	1236	135	DIAG GETF
DATAPR	1649	1236	400	GETF DI AG STORE PUNT CONSUB GETSYM
ASGNPR	1649	1236	70	RDLABL DIAG SYMBOL STORE GETC CKNAME
BDOPR	1649	1236	314	RDLABL DIAG STORE CKNAME CKIVC GETC GETSYM
CFIVOC	1649	1236	94	GETF DIAG STORE
CKIVC	----	1236	16	CFIVOC DIAG
CKNAME	----	1236	16	CFIVOC DIAG
IOSPR	1649	1236	1679	GETF DI AG SYMBOL STORE CKIVC RDLABL GETC STCHAR ENDDO BDOPR

Program	Common		Program	
Name	Labeled	Blank	Length	Externals
				OUTENT ARITH
ERBPR	1649	1236	83	CKIVC DIAG
MODMXR	1649	1236	1116	CNVT SYMBOL STORE PUNT
ASEMPR	1649	1236	434	GETC GETF DIAG STORE GETSYM RDLABL
TREE	1649	1236	1289	PUNT DIAG GETSYM MODMXR
ARAYSZ	1649	----	106	Q8PKUP Q8PREP
CPLOOP	1649	550	165	GETSYM ARAYSZ

FORTRAN 2.0A Compiler Programs: Phase B

Program	Common		Program	
Name	Labeled	Blank	Length	Externals
DUMMY	1739	1158	271	Q8PKUP Q8PREP GETSYM KSYMGN
FCMSTK	1739	1158	137	Q8PKUP Q8PREP KOUTPT
KCPART	----	----	49	Q8PKUP Q8PREP
KOUTPT	1739	1158	18	WRITE
KPCSTK	1739	1158	955	Q8PKUP Q8PREP KCPART GETSYM

Program Name	Common Labeled	Common Blank	Program Length	Externals
				DUMMY KOUTPT FCMSTK
KPC3PR	----	----	24	Q8PREP Q8PKUP KPCSTK
KSYMGN	1739	1158	72	Q8PKUP Q8PREP CNVT SYMBOL STORE
LABKPC	1739	1158	20	Q8PKUP Q8PREP KPCSTK
LABLER	1649	----	30	Q8PKUP Q8PREP KSYMGN
PUNT	1649	----	22	WRITE SKIPIT
SYMBOL	1649	3	157	GETSYM WRITE SKIPIT
TSALOC	1739	1158	139	Q8PKUP Q8PREP PUNT KSYMGN
ASSEM	1739	1158	103	KPCSTK LABKPC
BANANA	1739	1158	195	KPC3PR GETSYM LABKPC
BGINDO	1739	1158	265	PUNT LABLER GETSYM KPC3PR LABKPC
END	1739	1158	72	LABKPC INXRST KPC3PR ENTCOD GETSYM

Program Name	Common Labeled	Common Blank	Program Length	Externals
ENTCOD	1739	1158	169	KPC3PR LABKPC
HELEN	1739	1158	343	LABKPC KPC3PR ARAYSZ GETSYM SYMSCN
INXRST	1739	1158	20	KPC3PR
NOPROC	1739	1158	49	LABLER KPC3PR KOUTPT WRITE LABKPC
PHASEB	1739	1165	1109	LABLER LABKPC KPC3PR HELEN KPCSTK READIR TSALOC SUBFUN NOPROC ARITHR GETSYM ASUPER INXRST ENTCOD CGOTO BGINDO BANANA AFIDL PUNT ASSEM END
READIR	1739	1158	88	Q8PKUP Q8PREP PUNT READ KPC3PR LABKPC
SUBFUN	1739	1158	103	GETSYM LABLER

Program	Common		Program	
Name	Labeled	Blank	Length	Externals
ACP	1739	1158	1097	Q8PKUP Q8PREP PUNT INTRAM GETSYM KPC3PR KPCSTK TSALOC PARTSB FINK
AFIDL	1739	1158	90	Q8PKUP Q8PREP ASUPER KPC3PR
ASUPER	1739	1158	182	Q8PKUP Q8PREP PUNT SUBPR1 GETSYM SUBPR2 ACP
CGOTO	1739	1165	91	LABLER ASUPER KPC3PR KPCSTK LABKPC
FINK	1739	1158	181	KPCSTK LABLER KPC3PR LABKPC
INTRAM	1739	1158	473	PUNT KPCSTK KPC3PR TSALOC GETSYM LABKPC
PARTSB	1739	1158	174	Q8PKUP Q8PREP KPCSTK GETSYM KPC3PR SYMBOL STORE

Program Name	Common Labeled	Common Blank	Program Length	Externals
SUBPR1	1739	1158	62	Q8PKUP Q8PREP SUBPR3 TSALOC INTRAM KPC3PR
SUBPR2	1739	1158	141	Q8PKUP Q8PREP SUBPR3 KPC3PR GETSYM KPCSTK LABLER
SUBPR3	1739	1158	71	Q8PKUP Q8PREP ACP PARTSB
ARITHR	1739	1165	447	GETSYM SUBPR1 KPCSTK ASUPER KPC3PR

FORTRAN 2.0A Compiler Programs: Phase C

Program Name	Common Labeled	Common Blank	Program Length	Externals
BKDWN	2993	1148	95	------
BLDUP	2993	1148	67	------
BSS	2993	1148	30	BLDUP
CHKWD	2993	1148	382	GETSYM
CHOP	2993	1148	544	GETSYM
CL12	2993	1148	185	GETSYM CHOP BLDUP
CON	2993	1148	55	BLDUP
COUNT	2993	1148	23	------
DATAST	2993	1148	167	GETSYM BLDUP INOUT

Program	Common		Program	
Name	Labeled	Blank	Length	Externals
GETSYM	2993	1148	164	WRITE READ
INOUT	2993	1148	111	REED BKDWN WRITE
IXOPT	2993	1148	315	CHKWD BKDWN QXLD GETSYM
PHASEC	2993	1148	926	WRITE READ SYMSCN REED DATAST GETSYM CHOP LABEL BLDUP INOUT BSS COUNT BKDWN CHKWD LABIN IXOPT CON CL12 QXLD SKIP
LABEL	2993	1148	34	BLDUP INOUT
LABIN	2993	1148	102	REED LABEL
QXLD	2993	1148	144	Q8PKUP Q8PREP CHOP BLDUP INOUT COUNT
REED	2993	1148	93	READ
SKIP	2993	1148	86	CHOP BLDUP

Program Name	Common Labeled	Common Blank	Program Length	Externals
				INOUT COUNT
SYMSCN	2993	1148	28	GETSYM

FORTRAN 2.0A Compiler Programs: Phase D

Program Name	Common Labeled	Common Blank	Program Length	Externals
INDEX	1073	3095	28	Q8PKUP Q8PREP
NPUNCH	1073	3095	319	WRITE RESET READ
PHASE6	1073	3095	160	BEGIN READ INDEX AMT FINISH NPUNCH
AMT	----	----	9	AMOUT
AMOUT	1073	3095	1507	BKDWN ADMAX GETSYM INDEX COUNT LABOUT UNPUNC NP2OUT WRITE TABDEC NPUNCH SYMSCN
ADMAX	1073	3095	513	INDEX GETSYM TABDEC
BKDWN	1073	3095	105	INDEX
COUNT	1073	3095	23	------
LABOUT	1073	3095	223	Q8PKUP Q8PREP GETSYM UNPUNC RBPK

Program Name	Common		Program Length	Externals
	Labeled	Blank		
NP2OUT	1073	3095	47	RBPK COUNT WRITE
RBDX	1073	3095	60	------
RBPK	1073	3095	42	RBDX UNPUNC
TABDEC	1073	3095	132	SYMSCN
UNPUNC	1073	3095	22	RBDX NPUNCH
GETSYM	1073	3095	60	WRITE READ
SYMSCN	1073	3095	39	GETSYM
AMT	----	----	7	------
GETSYM	1073	3095	46	READ
IACON	1073	3095	88	------
IHCON	1073	3095	45	Q8PKUP Q8PREP
NWRITE	1073	3095	59	PACK WRITE
SYMSCN	1073	----	28	GETSYM
BEGINO	1073	3095	428	READ NWRITE IHCON GETSYM IACON NPUNCH SYMSCN WRITE
FINISH	1073	3095	410	NWRITE IHCON SYMSCN IACON NPUNCH

FORTRAN 2.0A Compiler Programs: Phase E

Program	Common		Program	
Name	Labeled	Blank	Length	Externals
INDEX	1073	2090	28	Q8PKUP Q8PREP
NPUNCH	1073	2090	319	WRITE RESET READ
PHASE6	1073	2090	160	BEGINO READ INDEX AMT FINISH NPUNCH
AMOUT	1073	2090	1513	BKDWN ADMAX GETSYM INDEX COUNT LABOUT UNPUNC NP2OUT WRITE TABDEC SETPRT CONV NPUNCH
ADMAX	1073	2090	513	INDEX GETSYM TABDEC
BKDWN	1073	2090	105	INDEX
COUNT	1073	2090	23	------
LABOUT	1073	2090	274	Q8PKUP Q8PREP GETSYM UNPUNC RBPK NWRITE IACON IHCON
NP2OUT	1073	2090	56	RBPK SETPRT COUNT WRITE

Program	Common		Program	
Name	Labeled	Blank	Length	Externals
RBDX	1073	2090	61	------
RBPK	1073	2090	42	RBDX UNPUNC
TABDEC	1073	2090	124	SYMSCN
UNPUNC	1073	2090	22	RBDX NPUNCH
GETSYM	1073	2090	77	WRITE READ
IACON	1073	2090	88	------
IHCON	1073	2090	44	Q8PKUP Q8PREP
NWRITE	1073	2090	59	PACK WRITE
SETPRT	1073	2090	379	IHCON IACON CONV NWRITE
BEGINO	1073	2090	329	READ NWRITE IHCON GETSYM IACON SETPRT NPUNCH SYMSCN
FINISH	1073	2090	367	NWRITE IHCON SYMSCN IACON NPUNCH

FORTRAN 2.0A Object Library Programs

Program	Common		Program	
Name	Labeled	Blank	Length	Externals
Q8IRFM	----	----	64	Q8PKUP Q8PREP Q8FS Q8TRAN

Program	Common		Program	
Name	Labeled	Blank	Length	Externals
Q8FS	----	----	464	Q8STP Q8PKUP Q8PREP Q8SKIP Q8FGET Q8FERM Q8RWBU Q8FPUT
Q8TRAN	----	----	1741	Q8STP Q8PKUP Q8PREP Q8FS Q8RWBU Q8MOVE Q8FERM Q8EXP9 Q8EXP1

4.3.7 MASS STORAGE FORTRAN 2.0A OBJECT LIBRARY PROGRAM ENTRY POINTS AND EXTERNALS

	Program Name	Entry Points	Externals
1.	FTN	FTN	
2.	Q8QINI	Q8QINI Q8UNIT Q8SKIP	Q8ERRM Q8INTB Q8EREM Q8QTOM Q8CMPO Q8MAGT Q8QEND Q8IFRM Q8IGP Q8CMP1 Q8QUN2 Q8DFIN
3.	Q8QEND	Q8QEND	Q8CMP1 Q8QUN1 Q8QUN2 Q8UNIT Q8DFAD
4.	Q8CMP	Q8CMP0 Q8CMP1 Q8DFAD Q8QENS	Q8EREM Q8BEGB Q8LOCB Q8CLRB

	Program Name	Entry Points	Externals
			Q8RINT Q8EOTT
5.	Q8RWBU	Q8BINB Q8LOCB Q8RWBU Q8INTB Q8BEGB Q8CLRB Q8RINT Q8IBUF	Q8CMP1 Q8EREM
6.	Q8ERRM	Q8ERRM Q8FERM Q8EREM	Q8LOCF Q8LOCB
7.	Q8DFIO	Q8DFNF Q8DFIN	Q8ERRM Q8EREM Q8QINI Q8DFAD Q8QENS
8.	Q8QX	Q8QTOM Q8QTRM Q8QX Q8MOVE Q8QY	Q8IFRM Q8BINB Q8QUN1 Q8UNIT
9.	Q8QUNI	Q8QUN1 Q8QUN2 Q8QUN3	Q8EREM
10.	Q8FGET	Q8FGET Q8FPUT Q8IGP Q8LOCF	
11.	Q8MAGT	Q8MAGT Q8EOTT	Q8EREM Q8QUN2 Q8COMI Q8QWND
12.	TAPCON	Q8QBCK Q8QFLE Q8QWND EOF	Q8EREM Q8CMP0 Q8CMP1 Q8QUN1 Q8QUN2 Q8QUN3 Q8IBUF
13.	IOCK	IOCK	Q8QUN1 Q8QUN3

	Program Name	Entry Points	Externals
14.	PSSTOP	Q8PSE Q8PSEN Q8STP Q8STPN Q8COMI	Q8PAND
15.	Q8PAND	Q8PAND	
16.	Q8EXP9	Q8EXP9 Q8EXPT Q8EXP2	FLOT
17.	Q8EXP1	Q8EXP1	FLOT Q8EXPT Q8EXP2
18.	Q8IFRM	Q8IFRM	Q8FS Q8TRAN Q8PKUP Q8PREP
19.	Q8FS	Q8FS	Q8SKIP Q8FGET Q8FERM Q8RWBU Q8FPUT Q8STP Q8PKUP Q8PREP
20.	Q8TRAN	Q8TRAN	Q8FS Q8RWBU Q8FERM Q8EXP9 Q8EXP1 Q8STP Q8PKUP Q8PREP Q8MOVE
21.	Q8AB	Q8AB ABS	FLOT
22.	SIGN	Q8SG SIGN	FLOT
23.	EXPPRG	EXP	FLOT
24.	SQRTF	SQRT	FLOT
25.	LNUPRG	ALOG	FLOT
26.	TANH	TANH	EXP FLOT

	Program Name	Entry Points	Externals
27.	SINCOS	SIN COS	FLOT
28.	ARCTPG	ATAN	FLOT
29.	Q8EXPN	Q8QF2I Q8QI2F Q8QF2F RETAD QSAVE	FLOT ALOG EXP
30.	FLOAT	FLOT	
31.	Q8PRMS	Q8PREP Q8PKUP	
32.	IFALT	IFALT	
33.	FXFL	Q8QFIX Q8FX Q8FLT Q8FLOT IFIX FLOAT	FLOT

4.3.8 MASS STORAGE FORTRAN 2.0B RELEASE TAPE FORMATS

FORTRAN 2.0B Installation Tape Format

The installation tape has the following format for magnetic tape. For paper tape, *K,I6,P8 is replaced by *K,I2,P8 and there is a*U at the end of each physical tape.

```
*K,I6,P8
*P
          FTN
          GOA
          CFIVOC
          CKNAME
          CNVT
          CONV
          DIAG
          EXP9
          FLOAT
          GETC
          GETF
          GETSYM
          GPUT
          IGETCF
          IOPRBA
          PACK
          Q8PRMS
          RDLABL
          STORE
          SYMBOL
          ENDDO
          GNST
          OPTION
          OUTENT
          PHASEA
          PLABEL
          STCHAR
          TYPE
          LOCLA1
          DUMYA1
          Q8QBDS
          ENDLOC
*T
*K,I8
*N,FORTA1,,,B
*K,I6,P8
*P
          FTN
          GOA
          CFIVOC
          CKNAME
          CNVT
          CONV
          DIAG
          EXP9
          FLOAT
          GETC
          GETF
          GETSYM
          GPUT
          IGETCF
          IOPRBA
          PACK
          Q8PRMS
          RDLABL
          STORF
          SYMBOL
          ENDDO
          GNST
          OPTION
          OUTENT
          PHASEA
          PLABEL
          STCHAR
          TYPE
          LOCLA2
          DUMYA2
          BYEQPR
          CHECKF
          COMNPR
          CONSUB
          DATAPR
          DIMPR
          EXRLPR
          FGETC
          FORK
          PEQVS
          PRNTNM
          SUBPPR
          SYMSCN
          TYPEPR
          ENDLOC
*T
*K,I8
*N,FORTA2,,,B
*K,I6,P8
*P
          FTN
          GOA
          CFIVOC
          CKNAME
          CNVT
          CONV
          DIAG
          EXP9
          FLOAT
          GETC
          GETF
          GETSYM
          GPUT
          IGETCF
          IOPRBA
          PACK
          Q8PRMS
          RDLABL
          STORE
          SYMBOL
          ENDDO
          GNST
          OPTION
          OUTENT
          PHASEA
          PLABEL
          STCHAR
          TYPE
          LOCLA3
          DUMYA3
          ARAYSZ
          ASEMPR
          ASGNPR
          BDOPR
          CHECKF
          CKIVC
          CONSUB
          CPLOOP
          DATAPR
          FGETC
          FORK
          ERBPR
          MODMXR
          PUNT
          ENDLOC
*T
*K,I8
*N,FORTA3,,,B
*K,I6,P8
*P
          FTN
          GOA
          CFIVOC
          CKNAME
          CNVT
          CONV
          DIAG
          EXP9
          FLOAT
          GETC
          GETF
          GETSYM
          GPUT
          IGETCF
          IOPRBA
          PACK
          Q8PRMS
          RDLABL
          STORE
          SYMBOL
          ENDDO
          GNST
          OPTION
          OUTENT
          PHASEA
          PLABEL
```

```
        STCHAR
        TYPE
        LOCLA4
        DUMYA4
        ARITH
        SUBSCR
        TREE
        ENDLOC
*T
*K,I8
*N,FORTA4,,,B
*K,I6,P8
*P
        FTN
        GOA
        CFIVOC
        CKNAME
        CNVT
        CONV
        DIAG
        EXP9
        FLOAT
        GETC
        GETF
        GETSYM
        GPUT
        IGETCF
        IOPRBA
        PACK
        Q8PRMS
        RDLABL
        STORE
        SYMBOL
        ENDDO
        GNST
        OPTION
        OUTENT
        PHASEA
        PLABEL
        STCHAR
        TYPE
        LOCLA5
        DUMYA5
        BDOPR
        CKIVC
        IOSPR
        ENDLOC
```

```
*T
*K,I8
*N,FORTA5,,,B
*K,I6,P8
*P
          FTN
          GOB
          CNVT
          DUMMY
          FCMSTK
          GETSYM
          IOPRBB
          KCPART
          KOUTPT
          KPCSTK
          KPC3PR
          KSYMGN
          LABKPC
          LABLER
          PUNT
          Q8PRMS
          STORE
          SYMBOL
          TSALOC
          ARAYSZ
          ASSEM
          BANANA
          BGINDO
          END
          ENTCOD
          HELEN
          INXRST
          NOPROC
          PHASEB
          READIR
          SUBFUN
          SYMSCN
          ACP
          AFIDL
          ASUPER
          CGOTO
          FINK
          INTRAM
          PARTSB
          SUBPR1
          SUBPR2
          SUBPR3
          ARITHR
          ENDLOC
```

```
*T
*K,I8
*N,FORTB1,,,B
*K,I6,P8
*P
          FTN
          GOC
          BKDWN
          BLDUP
          BSS
          CHKWD
          CHOP
          CL12
          CON
          COUNT
          DATAST
          GETSYM
          INOUT
          IOPRBC
          IXOPT
          LABEL
          LABIN
          PHASEC
          Q8PRMS
          QXLD
          REED
          SKIP
          SYMSCN
          ENDLOC
*T
*K,I8
*N,FORTC1,,,B
*K,I6,P8
*P
          FTN
          GOOD
          AMOUT
          ADMAX
          BEGINO
          BKDWN
          COUNT
          FINISH
          GETSYM
          IACON
          IHCON
          INDEX
          IOPRBD
          LABOUT
          NP2OUT
```

```
            NPUNCH
            NWRITE
            PACK
            PHASE6
            Q8PRMS
            RBDX
            RBPK
            SYMSCN
            TABDEC
            UNPUNC
            ENDLOC
*T
*K,I8
*N,FORTD1,,,B
*K,I6,P8
*P
            FTN
            GOE
            AMOUT
            ADMAX
            BEGINO
            BKDWN
            CONV
            COUNT
            FINISH
            GETSYM
            IACON
            IHCON
            INDEX
            IOPRBD
            LABOUT
            NP2OUT
            NPUNCH
            NWRITE
            PACK
            PHASE6
            Q8PRMS
            RBDX
            RBPK
            SETPRT
            SYMSCN
            TABDEC
            UNPUNC
            ENDLOC
*T
*K,I8
*N,FORTE1,,,B
*K,I6,P8
```

```
*L, FTN
          FTN
*L, Q8IFRM
          Q8IFRM
*L, Q8FS
          Q8FS
*L, Q8TRAN
          Q8TRAN
*L, FLOT
          FLOAT
*L, Q8QINI
          Q8QINI
*L, Q8QEND
          Q8QEND
*L, Q8CMP1
          Q8CMP
*L, Q8RWBU
          Q8RWBU
*L, Q8ERRM
          Q8ERRM
*L, Q8DFNF
          Q8DFIO
*L, Q8QX
          Q8QX
*L, Q8QUNI
          Q8QUNI
*L, Q8FGET
          Q8FGET
*L, Q8MAGT
          Q8MAGT
*L, Q8QBCK
          TAPCON
*L, IOCK
          IOCK
*L, Q8PSE
          PSSTOP
*L, Q8PAND
          Q8PAND
*L, Q8EXP9
          Q8EXP9
*L, Q8EXP1
          Q8EXP1
*L, Q8AB
          Q8AB
*L, SIGN
          SIGN
*L, EXP
          EXPPRG
```

```
*L, SQRT
          SQRTF
*L, ALOG
          LNUPRG
*L, TANH
          TANH
*L, SIN
          SINCOS
*L, ATAN
          ARCTPG
*L, QSAVE
          Q8EXPN
*L, IFALT
          IFALT
*L, Q8FX
          FXFL
*L, Q8PREP
          Q8PRMS
*U
```

FORTRAN 2.0B COSY Source Magnetic Tape

The FORTRAN 2.0B source tape is in COSY, compressed, format. The programs are arranged in the following order:

1. Phase A programs written in FORTRAN
2. Phase B programs written in FORTRAN
3. Phase C programs written in FORTRAN
4. Phase D programs written in FORTRAN
5. Phase E programs written in FORTRAN
6. Phases A, B, C, D, and E programs in assembly language
7. Object library programs written in FORTRAN
8. Object library programs written in assembly language

To assemble or compile a program, convert from COSY format into Hollerith format and then work with the Hollerith tape. Following is a list of the FORTRAN routine sequence numbers and COSY deck names.

COSY Deck Identifier	Program Name	COSY Deck Name	Phases
A1	CNVT	WCNVT	A1, A2, A3, A4, A5, B1
A2	GPUT	WGPUT	A1, A2, A3, A4, A5
A3	†SYMBOL	WSMBL1	A1, A2, A3, A4, A5

†Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program Name	COSY Deck Name	Phases
A4	GETF	WGETF1	A1, A2, A3, A4, A5
A5	GNST	WGNST	A1, A2, A3, A4, A5
A6	OUTENT	WOTENT	A1, A2, A3, A4, A5
A7	PHASEA	WPHSEA	A1, A2, A3, A4, A5
A8	PLABEL	WPLBEL	A1, A2, A3, A4, A5
A9	Q8QBDS	WQ8QBS	A1
A10	RDLABL	WRLABL	A1, A2, A3, A4, A5
A11	STCHAR	WSCHAR	A1, A2, A3, A4, A5
A12	TYPE	WTYPE	A1, A2, A3, A4, A5
A13	ARITH	WARITH	A4
A14	COMNPR	WCMNPR	A2
A15	DIMPR	WDIMPR	A2
A16	SUBSCR	WSBSCR	A4
A17	TYPEPR	WTYPPR	A2
A18	BYEQPR	WBYEQ	A2
A19	CHECKF	WCKF	A2, A3
A20	FGETC	WFGETC	A2, A3
A21	FORK	WFORK	A2, A3
A23	SUBPPR	WSUB	A2
A234	EXRLPR	WEXRL	A2
A245	PEQVS	WPEQVS	A2
A256	PRNTNM	WPRTNM	A2
A267	†PUNT	WPUNT1	A3
A278	†SYMSCN	WSYMS1	A2, B1
A289	ENDDO	WENDDO	A1, A2, A3, A4, A5
A30	CONSUB	WCONSB	A2, A3
A31	DATAPR	WDATA	A2, A3
A32	ASGNPR	WASGN	A3
A33	BDOPR	WBDOPR	A3, A5
A34	CFIVOC	WCFVOC	A1, A2, A3, A4, A5

†Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program Name	COSY Deck Name	Phases
A345	CKIVC	WCKVC	A3, A5
A356	CKNAME	WCKNAM	A1, A2, A3, A4, A5
A367	IOSPR	WIOSPR	A5
A38	ERBPR	WERBPR	A3
A389	MODMXR	WMODX	A3
A40	ASEMPR	WASEMP	A3
A41	TREE	WTREE	A4
A42	ARAYSZ	WARAY	A3, B1
A43	CPOLLP	WLOOP	A3
A50	DUMMY	WDUMMY	B1
A51	FCMSTK	WFCMK	B1
A52	KCPART	WKCPRT	B1
A53	KOUTPT	WKOTPT	B1
A54	KPCSTK	WKPCK	B1
A55	KPC3PR	WKPC3	B1
A56	KSYMGN	WKSYM	B1
A57	LABKPC	WLBKPC	B1
A58	LABLER	WLABLR	B1
A59	†PUNT	WPUNT2	B1
A60	†SYMBOL	WSMBL2	B1
A61	TSALOC	WTSLOC	B1
A62	ASSEM	WASSEM	B1
A63	BANANA	WBANAN	B1
A64	BGINDO	WBGNDO	B1
A65	END	WEND	B1
A66	ENTCOD	WENTCD	B1
A67	HELEN	WHELEN	B1
A68	INXRST	WXRST	B1
A69	NOPROC	WNOPR	B1
A70	PHASEB	WPHSEB	B1

†Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program Name	COSY Deck Name	Phases
A71	READIR	WRDIR	B1
A72	SUBFUN	WSUBFN	B1
A73	ACP	WACP	B1
A74	AFIDL	WAFIDL	B1
A75	ASUPER	WASPER	B1
A76	CGOTO	WCGOTO	B1
A77	FINK	WFINK	B1
A78	INTRAM	WTRAM	B1
A79	PARTSB	WPRTSB	B1
A80	SUBPR1	WSUB1	B1
A81	SUBPR2	WSUB2	B1
A82	SUBPR3	WSUB3	B1
A83	ARITHR	WRITHR	B1
B1	†BKDWN	WBKDN1	C1
B2	BLDUP	WBLDUP	C1
B3	BSS	WBSS	C1
B4	CHKWD	WCHKWD	C1
B5	CHOP	WCHOP	C1
B6	CL12	WCL12	C1
B7	CON	WCON	C1
B8	†COUNT	WCNT1	C1
B9	DATAST	WDATST	C1
B10	†GETSYM	WGSYM1	C1, D1, E1
B11	INOUT	WINOUT	C1
B12	IXOPT	WIXOPT	C1
B13	PHASEC	WPHSEC	C1
B14	LABEL	WLABEL	C1
B15	LABIN	WLABIN	C1
B16	QXLD	WQXLD	C1
B17	REED	WREED	C1

†Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program Name	COSY Deck Name	Phases
B18	†SKIP	WSKIP	C1
B19	†SYMSCN	WSYMS2	C1
B20	†AMOUT	WAMOT1	D1
B21	†ADMAX	WMAX1	D1
B22	†BEGINO	WGINO1	D1
B23	†BKDWN	WBKDN2	D1
B24	†COUNT	WCNT2	D1
B25	†FINISH	WFIN1	D1
B26	†IACON	WIACN1	D1
B27	†IHCON	WIHCN1	D1
B28	†INDEX	WDEX1	D1
B29	†LABOUT	WLABT1	D1
B30	†NP2OUT	WNP2T1	D1
B31	†NPUNCH	WNPUN1	D1
B32	†NWRITE	WNRIT1	D1
B33	†PHASE6	WPHS61	D1
B34	†RBDX	WRBDX1	D1
B35	†RBPK	WRBPK1	D1
B36	†SYMSCN	WSYMS3	D1, E1
B37	†TABDEC	WDEC1	D1
B38	†UNPUNC	WUNPC1	D1
B39	†AMOUT	WAMOT2	E1
B40	†ADMAX	WMAX2	E1
B41	†BEGINO	WGINO2	E1
B42	†BKDWN	WBKDN2	E1
B43	†COUNT	WCNT3	E1
B44	†FINISH	WFIN2	E1
B45	†IACON	WIACN2	E1
B46	†IHCON	WIHCN2	E1
B47	†INDEX	WDEX2	E1

†Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program Name	COSY Deck Name	Phases
B48	†LABOUT	WLABT2	E1
B49	†NP2OUT	WNP2T2	E1
B50	†NPUNCH	WNPUN2	E1
B51	†NWRITE	WNRIT2	E1
B52	†PHASE6	WHPS62	E1
B53	†RBDX	WRBDX2	E1
B54	†RBPK	WRBPK2	E1
B55	†SETPRT	WSPRT	E1
B56	†TABDEC	WDEC2	E1
B57	†UNPUNC	WUNPC2	E1
B60	FTN	WFTNB	A1, A2, A3, A4, A5. B1, C1, D1, E1
B61	GOA	WGOA	A1, A2, A3, A4, A5
B62	†CONV	WCONV1	A1, A2, A3, A4, A5
B63	DIAG	WDIAG	A1, A2, A3, A4, A5
B64	EXP9	WEXP9	A1, A2, A3, A4, A5
B65	FLOAT	WFLOAT	A1, A2, A3, A4, A5
B66	GETSYM	WGSYM2	A1, A2, A3, A4, A5, B1
B67	IOPRBA	WIOPRA	A1, A2, A3, A4, A5
B68	PACK	WPACK	A1, A2, A3, A4, A5, D1, E1
B69	Q8PRMS	WQ8P	A1, A2, A3, A4, A5, B1, C1, D1, E1
B70	STORE	WSTORE	A1, A2, A3, A4, A5, B1
B71	LOCLA1	WLA1	A1
B72	DUMYA1	WDA1	A1
B73	GETC	WGETC1	A1, A2, A3, A4, A5
B74	IGETCF	WIGTCF	A1, A2, A3, A4, A5
B75	OPTION	WOPT	A1, A2, A3, A4, A5
B76	ENDLOC	WELOC	A1, A2, A3, A4, A5, B1, C1, D1, E1
B77	LOCLA2	WLA2	A2
B78	DUMYA2	WDA2	A2
B79	LOCLA3	WLA3	A3

†Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

COSY Deck Identifier	Program Name	COSY Deck Name	Phases
B80	DUMYA3	WDA3	A3
B81	LOCLA4	WLA4	A4
B82	DUMYA4	WDA4	A4
B83	LOCLA5	WLA5	A5
B84	DUMYA5	WDA5	A5
B85	GOB	WGOB	B1
B86	IOPRBB	WIOPRB	B1
B87	GOC	WGOC	C1
B88	IOPRBC	WIOPRC	C1
B89	GOOD	WGOOD	D1
B90	IOPROBD	WIOPRD	D1, E1
B91	GOE	WGOE	E1
B92	†CONV	WCONV2	E1

Object Library Programs written in FORTRAN

Deck Identifier	Program	Deck Name
Q01	Q8IFRM	Q8IFRM
Q02	Q8FS	Q8FS
Q03	Q8TRAN	Q8TRAN

Object Library Programs written in assembly language

Deck Identifier	Program	Deck Name
Q04	†FLOAT	FLOAT2
Q05	Q8QINI	Q8QINI
Q06	Q8QEND	Q8QEND
Q07	Q8CMP	Q8CMP
Q08	Q8RWBU	Q8RWBU

†Indicates that there is at least one other different program with the same name and care should be taken to make sure the correct program is selected.

Deck Identifier	Program	Deck Name
Q09	Q8ERRM	Q8ERRM
Q10	Q8DFIO	Q8DFIO
Q11	Q8QX	Q8QX
Q12	Q8QUNI	Q8QUNI
Q13	Q8FGET	Q8FGET
Q14	Q8MAGT	Q8MAGT
Q15	TAPCON	TAPCON
Q16	IOCK	IOCK
Q17	PSSTOP	PSSTOP
Q18	Q8PAND	Q8PAND
Q19	Q8EXP9	Q8EXP9
Q20	Q8EXP1	Q8EXP1
Q21	Q8AB	Q8AB
Q22	SIGN	SIGN
Q23	EXPPRG	EXPPRG
Q24	SQRTF	SQRTF
Q25	LNUPRG	LNUPRG
Q26	TANH	TANH
Q27	SINCOS	SINCOS
Q28	ARCTPG	ARCTPG
Q29	Q8EXPN	Q8EXPN
Q30	IFALT	IFALT
Q31	FXFL	FXFL

4.3.9 MASS STORAGE FORTRAN 2.0B COMPILER PROGRAM ORDER

The compiler consists of four passes over the source code or its equivalent, accomplished in four phases called A, B, C, and D/E. (The fourth pass is performed by either Phase D or Phase E, depending upon whether the user wants to use an assembly language listing output.) Each phase consists of a root which is core-resident throughout the phase, and zero or more local subroutine groups which share the same core area and are read from disc as needed.

Phase A reads the source input, converts it to statements expressed in an internal code, and assigns a statement number to the statement.

Phase B reads the output of Phase A and generates pseudo code from it. (Pseudo code is similar to assembler input except that the index to be used in an indexed instruction and the addressing mode are not specified.)

Phase C and D/E are a two-pass assembler. The output from Phase B is read. Index registers are optimally assigned. One word relative addressing is maximized. Relocatable binary output and an assembly listing are produced.

Pass	Root	Local
FORTA1	A	1
FORTA2	A	2
FORTA3	A	3
FORTA4	A	4
FORTA5	A	5
FORTB1	B	
FORTC1	C	
FORTD1	D	
FORTE1	E	

Pass A

Root:

FTN
GOA
CFIVOC
CKNAME
CNVT
CONV
DIAG
EXP9
FLOAT
GETC
GETF
GETSYM
GPUT
IGETCF
IOPRBA
PACK
Q8PRMS
RDLABL
STORE
SYMBOL
ENDDO
GNST

OPTION
OUTENT
PHASEA
PLABEL
STCHAR
TYPE

Local 1:

LOCLA1
DUMYA1
Q8QBDS
ENDLOC

Local 2:

LOCLA2
DUMYA2
BYEQPR
CHECKF
COMNPR
CONSUB
DATAPR
DIMPR
EXRLPR
FGETC
FORK
PEQVS
PRNTNM
SUBPPR
SYMSCN
TYPEPR
ENDLOC

Local 3:

LOCLA3
DUMYA3
ARAYSZ
ASEMPR
ASGNPR
BDOPR
CHECKF
CKIVC
CONSUB
CPLOOP
DATAPR
FGETC
FORK
ERBPR

MODMXR
PUNT
ENDLOC

Local 4:

LOCLA4
DUMYA4
ARITH
SUBSCR
TREE
ENDLOC

Local 5:

LOCLA5
DUMYA5
BDOPR
CKIVC
IOSPR
ENDLOC

<u>Pass B</u>

Root:

FTN
GOB
CNVT
DUMMY
FCMSTK
GETSYM
IOPRBB
KCPART
KOUTPT
KPCSTK
KPC3PR
KSYMGN
LABKPC
LABLER
PUNT
Q8PRMS
STORE
SYMBOL
TSALOC
ARAYSZ
ASSEM
BANANA
BGINDO
END

ENTCOD
HELEN
INXRST
NOPROC
PHASEB
READIR
SUBFUN
SYMSCN
ACP
AFIDL
ASUPER
CGOTO
FINK
INTRAM
PARTSB
SUBPR1
SUBPR2
SUBPR3
ARITHR
ENDLOC

Pass C

Root:

FTN
GOC
BKDWN
BLDUP
BSS
CHKWD
CHOP
CL12
CON
COUNT
DATAST
GETSYM
INOUT
IOPRBC
IXOPT
LABEL
LABIN
PHASEC
Q8PRMS
QXLD
REED
SKIP
SYMSCN
ENDLOC

Pass D

Root:

FTN
GOOD
AMOUT
ADMAX
BEGINO
BKDWN
COUNT
FINISH
GETSYM
IACON
IHCON
INDEX
IOPRBD
LABOUT
NP2OUT
NPUNCH
NWRITE
PACK
PHASE6
Q8PRMS
RBDX
RBPK
SYMSCN
TABDEC
UNPUNC
ENDLOC

Pass E

Root:

FTN
GOE
AMOUT
ADMAX
BEGINO
BKDWN
CONV
COUNT
FINISH
GETSYM
IACON
IHCON
INDEX
IOPRBD
LABOUT

NP2OUT
NPUNCH
NWRITE
PACK
PHASE6
Q8PRMS
RBDX
RBPK
SETPRT
SYMSCN
TABDEC
UNPUNC
ENDLOC

4.3.10 MASS STORAGE FORTRAN 2.0B COMPILER PROGRAM LENGTHS AND EXTERNALS

Program Name	Labeled Common	Blank Common	Program Length	Externals
CNVT	1641	3	62	
GPUT	1641	1236	41	
SYMBOL	1641	3	162	GETSYM WRITE SKIPIT
GETF	1641	1236	760	GETC GPUT DIAG EXP9 CNVT SYMBOL
GNST	1641	1236	446	CONV PACK WRITE IGETCF READ STCHAR EXIT DIAG SKIPIT
OUTENT	1641	1236	52	Q8PKUP Q8PREP WRITE
PHASEA	1641	1236	1259	WRITE GNST PLABEL TYPE

Program Name	Labeled Common	Blank Common	Program Length	Externals
				DIAG OUTENT PEQVS DIMPR COMNPR TYPEPR SUBPPR GETF STORE BYEQPR EXRLPR GETC DATAPR CHECKF ARITH GETSYM SYMBOL ASGNPR RDLABL ENDDO CPLOOP SKIPIT ERBPR IOSPR BDOPR STCHAR ASEMPR
PLABEL	1641	1236	86	RDLABL STORE DIAG
Q8QBDS	1641	0	0	
RDLABL	1641	1236	129	GETC DIAG CNVT SYMBOL
STCHAR	0	0	50	Q8PKUP Q8PREP
TYPE	1641	1236	510	Q8PKUP Q8PREP GETC GETF
ARITH	1641	1236	1669	LOCAL GETF

Program	Labeled	Blank	Program	
Name	Common	Common	Length	Externals
				DIAG PUNT TREE GETSYM SUBSCR STORE GETC SYMBOL
COMNPR	1641	1236	150	GETF DIMPR DIAG
DIMPR	1641	1236	391	GETF DIAG STORE GETSYM
SUBSCR	1641	1236	701	GETF DIAG STORE PUNT GETSYM SYMBOL
TYPEPR	1641	1236	23	DIMPR
BYEQPR	1641	1236	499	GETF DIAG STORE GETSYM
CHECKF	1641	1236	160	FORK DIAG FGETC
FGETC	1641	1236	31	Q8PKUP Q8PREP GETC STCHAR
FORK	0	0	370	Q8PKUP Q8PREP FGETC
SUBPPR	1641	1236	181	GETF DIAG STORE
EXRLPR	1641	1236	94	GETF DIAG STORE

Program Name	Labeled Common	Blank Common	Program Length	Externals
PEQVS	1641	1236	991	SYMSCN GETSYM PRNTNM DIAG
PRNTNM	1641	0	141	Q8PKUP Q8PREP GETSYM WRITE
PUNT	0	1236	56	DIAG READ IGETCF SKIPIT
SYMSCN	1553	0	28	GETSYM
ENDDO	1641	1236	261	GETSYM OUTENT DIAG
CONSUB	1641	1236	135	DIAG GETF
DATAPR	1641	1236	400	GETF DIAG STORE PUNT CONSUB GETSYM
ASGNPR	1641	1236	70	RDLABL DIAG SYMBOL STORE GETC CKNAME
BDOPR	1641	1236	314	RDLABL DIAG STORE CKNAME CKIVC GETC GETSYM
CFIVOC	1641	1236	94	GETF DIAG STORE
CKIVC	0	1236	16	CFIVOC DIAG

Program Name	Labeled Common	Blank Common	Program Length	Externals
CKNAME	0	1236	16	CFIVOC DIAG
IOSPR	1641	1236	1699	LOCAL GETF DIAG SYMBOL STORE CKIVC RDLABL GETC STCHAR ENDDO BDOPR OUTENT
ERBPR	1641	1236	83	CKIVC DIAG
MODMXR	1641	1236	1116	CNVT SYMBOL STORE PUNT GETSYM
ASEMPR	1641	1236	434	GETC GETF DIAG STORE GETSYM RDLABL
TREE	1641	1236	1280	PUNT DIAG GETSYM
ARAYSZ	1641	0	106	Q8PKUP Q8PREP
CPLOOP	1641	550	165	GETSYM ARAYSZ
DUMMY	1547	1158	271	Q8PKUP Q8PREP GETSYM KSYMGN WRITE SKIPIT
FCMSTK	1547	1158	137	Q8PKUP Q8PREP KOUTPT

Program Name	Labeled Common	Blank Common	Program Length	Externals
KCPART	0	0	49	Q8PKUP Q8PREP
KOUTPT	1547	1158	18	WRITE
KPCSTK	1547	1158	955	Q8PKUP Q8PREP KCPART GETSYM DUMMY KOUTPT FCMSTK
KPC3PR	0	0	24	Q8PKUP Q8PREP KPCSTK
KSYMGN	1547	1158	72	Q8PKUP Q8PREP CNVT SYMBOL STORE
LABKPC	1547	1158	20	Q8PKUP Q8PREP KPCSTK
LABLER	1649	0	30	Q8PKUP Q8PREP KSYMGN
PUNT	1649	0	22	WRITE SKIPIT
SYMBOL	1649	3	157	GETSYM WRITE SKIPIT
TSALOC	1547	1158	139	Q8PKUP Q8PREP PUNT KSYMGN
ASSEM	1547	1158	103	KPCSTK LABKPC
BANANA	1547	1158	195	KPC3PR GETSYM LABKPC
BGINDO	1547	1158	265	PUNT LABLER

Program Name	Labeled Common	Blank Common	Program Length	Externals
				GETSYM
				KPC3PR
				LABKPC
END	1547	1158	72	LABKPC
				INXRST
				KPC3PR
				ENTCOD
				GETSYM
ENTCOD	1547	1158	169	KPC3PR
				LABKPC
HELEN	1547	1158	343	LABKPC
				KPC3PR
				ARAYSZ
				GETSYM
				SYMSCN
INXRST	1547	1158	20	KPC3PR
NOPROC	1547	1158	49	LABLER
				KPC3PR
				KOUTPT
				WRITE
				LABKPC
PHASEB	1547	1158	1109	LABLER
				LABKPC
				KPC3PR
				HELEN
				KPCSTK
				READIR
				TSALOC
				SUBFUN
				NOPROC
				ARITHR
				GETSYM
				ASUPER
				INXRST
				ENTCOD
				CGOTO
				BGINDO
				BANANA
				AFIDL
				PUNT
				ASSEM
				END

Program Name	Labeled Common	Blank Common	Program Length	Externals
READIR	1547	1158	88	Q8PKUP Q8PREP PUNT READ KPC3PR LABKPC
SUBFUN	1547	1158	103	GETSYM LABLER
ACP	1547	1158	1097	Q8PKUP Q8PREP PUNT INTRAM GETSYM KPC3PR KPCSTK TSALOC PARTSB FINK
AFIDL	1547	1158	90	Q8PKUP Q8PREP ASUPER KPC3PR
ASUPER	1547	1158	182	Q8PKUP Q8PREP PUNT SUBPR1 GETSYM SUBPR2 ACP
CGOTO	1547	1158	91	LABLER ASUPER KPC3PR KPCSTK LABKPC
FINK	1547	1158	181	KPCSTK LABLER KPC3PR LABKPC
INTRAM	1547	1158	473	PUNT KPCSTK KPC3PR TSALOC

Program Name	Labeled Common	Blank Common	Program Length	Externals
				GETSYM LABKPC
PARTSB	1547	1158	174	Q8PKUP Q8PREP KPCSTK GETSYM KPC3PR SYMBOL STORE
SUBPR1	1547	1158	62	Q8PKUP Q8PREP SUBPR3 TSALOC INTRAM KPC3PR
SUBPR2	1547	1158	141	Q8PKUP Q8PREP SUBPR3 KPC3PR GETSYM KPCSTK LABLER
SUBPR3	1547	1158	71	Q8PKUP Q8PREP ACP PARTSB
ARITHR	1547	1158	447	GETSYM SUBPR1 KPCSTK ASUPER KPC3PR
BKDWN	2993	1148	95	
BLDUP	2993	1148	67	
BSS	2993	1148	30	BLDUP
CHKWD	2993	1148	382	GETSYM
CHOP	2993	1148	544	GETSYM
CL12	2993	1148	185	GETSYM CHOP BLDUP
CON	2993	1148	55	BLDUP

Program Name	Labeled Common	Blank Common	Program Length	Externals
COUNT	2993	1148	23	
DATAST	2993	1148	167	GETSYM BLDUP INOUT
GETSYM	2993	0	164	WRITE READ
INOUT	2993	1148	111	REED BKDWN WRITE
IXOPT	2993	1148	315	CHKWD BKDWN QXLD GETSYM
PHASEC	2993	1148	847	WRITE READ SYMSCN REED DATAST GETSYM CHOP LABEL BLDUP INOUT BSS COUNT BKDWN CHKWD LABIN IXOPT CON CL12 QXLD SKIP
LABEL	2993	1148	34	BLDUP INOUT
LABIN	2993	1148	102	REED LABEL
QXLD	2993	1148	144	Q8PKUP Q8PREP CHOP BLDUP INOUT COUNT

Program Name	Labeled Common	Blank Common	Program Length	Externals
REED	2993	1148	93	READ
SKIP	2993	1148	86	CHOP BLDUP INOUT COUNT
SYMSCN	2993	1148	28	GETSYM
AMOUT	2993	1148	1498	BKDWN ADMAX GETSYM INDEX COUNT LABOUT UNPUNC NP2OUT WRITE TABDEC NPUNCH
ADMAX	2993	1148	513	INDEX GETSYM TABDEC
BEGINO	2993	1148	272	NWRITE IHCON GETSYM IACON NPUNCH
BKDWN	2993	1148	105	INDEX
COUNT	2993	1148	23	
FINISH	2993	1148	367	NWRITE IHCON SYMSCN IACON NPUNCH
IACON	2993	1148	88	
IHCON	2993	1148	45	Q8PKUP Q8PREP
INDEX	2993	1148	28	Q8PKUP Q8PREP
LABOUT	2993	1148	223	Q8PKUP Q8PREP GETSYM

Program Name	Labeled Common	Blank Common	Program Length	Externals
				UNPUNC RBPK
NP2OUT	2993	1148	47	RBPK COUNT WRITE
NPUNCH	2993	1148	319	WRITE RESET READ
NWRITE	2993	1148	59	PACK WRITE
PHASE6	2993	1148	156	BEGINO READ INDEX AMOUT FINISH NPUNCH
RBDX	2993	1148	60	
RBPK	2993	1148	42	RBDX UNPUNC
SYMSCN	2993	1148	28	GETSYM
TABDEC	2993	1148	132	SYMSCN
UNPUNC	2993	1148	22	RBDX NPUNCH
AMOUT	2993	2090	1513	BKDWN ADMAX GETSYM INDEX COUNT LABOUT UNPUNC NP2OUT WRITE TABDEC SETPRT CONV NPUNCH
ADMAX	2993	2090	513	INDEX GETSYM TABDEC

Program Name	Labeled Common	Blank Common	Program Length	Externals
BEGINO	2993	2090	321	NWRITE IHCON GETSYM IACON SETPRT NPUNCH SYMSCN
BKDWN	2993	2090	105	INDEX
COUNT	2993	2090	23	
FINISH	2993	2090	367	NWRITE IHCON SYMSCN IACON NPUNCH
IACON	2993	2090	88	
IHCON	2993	2090	44	Q8PKUP Q8PREP
INDEX	2993	2090	28	Q8PKUP Q8PREP
LABOUT	2993	2090	274	Q8PKUP Q8PREP GETSYM UNPUNC RBPK NWRITE IACON IHCON
NP2OUT	2993	2090	56	RBPK SETPRT COUNT WRITE
NPUNCH	2993	2090	319	WRITE RESET READ
NWRITE	2993	2090	59	PACK WRITE
PHASE6	2993	2090	156	BEGINO READ INDEX AMOUT

Program Name	Labeled Common	Blank Common	Program Length	Externals
				FINISH NPUNCH
RBDX	2993	2090	61	
RBPK	2993	2090	42	RBDX UNPUNC
SETPRT	2993	2090	379	IHCON IACON CONV NWRITE
TABDEC	2993	2090	124	SYMSCN
UNPUNC	2993	2090	22	RBDX NPUNCH

4.3.11 MASS STORAGE FORTRAN 2.0B OBJECT LIBRARY PROGRAM ENTRY POINTS AND EXTERNALS

	Program Name	Entry Points	Externals
1.	FTN	FTN	
2.	Q8QINI	Q8QINI Q8UNIT Q8SKIP	Q8ERRM Q8INTB Q8EREM Q8QTOM Q8CMPO Q8MAGT Q8QEND Q8IFRM Q8IGP Q8CMP1 Q8QUN2 Q8DFIN
3.	Q8QEND	Q8QEND	Q8CMP1 Q8QUN1 Q8QUN2 Q8UNIT Q8DFAD
4.	Q8CMP	Q8CMP0 Q8CMP1 Q8DFAD Q8QENS	Q8EREM Q8BEGB Q8LOCB Q8CLRB Q8RINT Q8EOTT

	Program Name	Entry Points	Externals
5.	Q8RWBU	Q8BINB Q8LOCB Q8RWBU Q8INTB Q8BEGB Q8CLRB Q8RINT Q8IBUF	Q8CMP1 Q8EREM
6.	Q8ERRM	Q8ERRM Q8FERM Q8EREM	Q8LOCF Q8LOCB
7.	Q8DFIO	Q8DFNF Q8DFIN	Q8ERRM Q8EREM Q8QINI Q8DFAD Q8QENS
8.	Q8QX	Q8QTOM Q8QTRM Q8QX Q8MOVE Q8QY	Q8IFRM Q8BINB Q8QUN1 Q8UNIT
9.	Q8QUNI	Q8QUN1 Q8QUN2 Q8QUN3	Q8EREM
10.	Q8FGET	Q8FGET Q8FPUT Q8IGP Q8LOCF	
11.	Q8MAGT	Q8MAGT Q8EOTT	Q8EREM Q8QUN2 Q8COMI Q8QWND
12.	TAPCON	Q8QBCK Q8QFLE Q8QWND EOF	Q8EREM Q8CMP0 Q8CMP1 Q8QUN1 Q8QUN2 Q8QUN3 Q8IBUF
13.	IOCK	IOCK	Q8QUN1 Q8QUN3
14.	PSSTOP	Q8PSE Q8PSEN	Q8PAND

	Program Name	Entry Points	Externals
		Q8STP Q8STPN Q8COMI	
15.	Q8PAND	Q8PAND	
16.	Q8EXP9	Q8EXP9 Q8EXP1 Q8EXP2	FLOT
17.	Q8EXP1	Q8EXP1	FLOT Q8EXPT Q8EXP2
18.	Q8IFRM	Q8IFRM	Q8FS Q8TRAN Q8PKUP Q8PREP
19.	Q8FS	Q8FS	Q8SKIP Q8FGET Q8FERM Q8RWBU Q8FPUT Q8STP Q8PKUP Q8PREP
20.	Q8TRAN	Q8TRAN	Q8FS Q8RWBU Q8FERM Q8EXP9 Q8EXP1 Q8STP Q8PKUP Q8PREP Q8MOVE
21.	Q8AB	Q8AB ABS	FLOT
22.	SIGN	Q8SG SIGN	FLOT
23.	EXPPRG	EXP	FLOT
24.	SQRTF	SQRT	FLOT
25.	LNUPRG	ALOG	FLOT
26.	TANH	TANH	EXP FLOT

	Program Name	Entry Points	Externals
27.	SINCOS	SIN COS	FLOT
28.	ARCTPG	ATAN	FLOT
29.	Q8EXPN	Q8QF2I Q8QI2F Q8QF2F RETAD QSAVE	FLOT ALOG EXP
30.	FLOAT	FLOT	
31.	Q8PRMS	Q8PREP Q8PKUP	
32.	IFALT	IFALT	
33.	FXFL	Q8QFIX Q8FX Q8FLT Q8FLOT IFIX FLOAT	FLOT

4.3.12 COSY 1.0 RELEASE TAPE FORMAT

The installation tape has the following format

COSY – RELOCATABLE BINARY

EOF

EOF

The COSY source tape has the following format

DECK – COSY

EOF

4.3.13 SYSTEM CONFIGURATOR RELEASE TAPE FORMAT

Installation Tapes

The System Configurator installation tape has the following format for magnetic tape. The paper tape format *K, I2, P8 replaces *K, I6, P8. Each deck name refers to a relocatable binary deck. Only the magnetic tape version is folllowed by two end-of-files.

```
*L, CONFIG
          CONFIG
*K, I6, P8
*P, F, GOCONF
          CONFIG
          GOCONF
          SCDKIO
          ERROR
          DCTOAS
          GETITM
          CALADR
          INCPTR
          GETFLE
          GO1A
          OPTCHK
          INPREC
          MESSGS
          SCNOPT
          INITAL
          CONVRT
          CONTRL
          CORECT
          INSINP
          SCNREC
*T
*K, I8
*N, CONF1A, , , B
*K, I6
*P, F, GO1B
          CONFIG
          GOCONF
          SCDKIO
          ERROR
          DCTOAS
          GETITM
          CALADR
          INCPTR
          GETFLE
          GO1B
          DEFINE
          PARCHK
          PAMCHK
          PARTIT
          SEARCH
          SCNREC
          INPREC
          CONTRL
          VALPRO
          VALCHK
          PICKUP
*T
*K,I8
*N,CONF1B,,,B
*K,I6
*P,F,GO1C
          CONFIG
          GOCONF
          SCDKIO
          ERROR
          DCTOAS
          GETITM
          CALADR
          INCPTR
          GETFLE
          GO1C
          SYSDAT
          SYSINS
          INSINP
          INCINS
          GETCHR
          STOCHR
          WRTMMR
          RDSKEL
          INITCM
          COMMNT
*T
*K,I8
*N,CONF1C,,,B
*K,I6
*P,F,GO1D
          CONFIG
          GOCONF
          SCDKIO
          ERROR
          DCTOAS
          GETITM
          CALADR
          INCPTR
          GETFLE
          GO1D
          SPECF1
          PARCHK
          BKCMVR
```

```
         SPCPAR
         SEARCH
         SCNREC
         CONTRL
         INPREC
         CORECK
         CORECT
         CONVRT
         INSINP
*T
*K,I8
*N,CONF1D,,,B
*K,I6
*P,F,GO1E
         CONFIG
         GOCONF
         SCDKIO
         ERROR
         DCTOAS
         GETITM
         CALADR
         INCPTR
         GETFLE
         GO1E
         SPECF2
         PAMCH2
         INSURT
         INCINS
         INSINP
         SEARCH
         CORECT
         SCNREC
         CONTRL
         INPREC
         CNVTNO
         PICKUP
*T
*K,I8
*N,CONF1E,,,B
*K,I6
*P,F,GO1F
         CONFIG
         GOCONF
         SCDKIO
         ERROR
         DCTOAS
         GETITM
         CALADR
         INCPTR
         GETFLE
```

```
          GO1F
          VARPRO
          RNGCHK
          SCNREC
          OORECT
          VALCHK
          INPREC
          CONTRL
          CNVTNO
          PICKUP
*T
*K,I8
*N,CONF1F,,,B
*K,I6
*P,F,GO2
          CONFIG
          GOCONF
          SCDKIO
          ERROR
          DCTOAS
          GETITM
          CALADR
          INCPTR
          GETFLE
          GO2
          PHASE2
          EQUIVA
          INSERT
          DELETE
          GETVAL
          CVTNUM
          GETNUM
          REDREC
          GETCHR
          GNSCHR
          STOCHR
          DECASC
          MMREAD
          OUTREC
          HICORE
          INTREG
          PICKUP
          P2NAM1
*T
*K,I8
*N,CONF2A,,,B
*K,I6
*P,F,GO2
          CONFIG
          GOCONF
          SCDKIO
          ERROR
          DCTOAS
          GETITM
          CALADR
          INCPTR
          GETFLE
          GO2
          PHASE2
          EQUIVA
          DELETE
          GETVAL
          CVTNUM
          GETNUM
          REDREC
          GETCHR
          GNSCHR
          STOCHR
          DECASC
          MMREAD
          OUTREC
          MSKTBL
          FTNLVL
          SCHSTK
          PICKUP
          P2NAM2
*T
*K, I8
*N, CONF2B, , , B
*K, I6
*P, F, GO2
          CONFIG
          GOCONF
          SCDKIO
          ERROR
          DCTOAS
          GETITM
          CALADR
          INCPTR
          GETFLE
          GO2
          PHASE2
          EQUIVA
          DELETE
          GETVAL
          CVTNUM
          GETNUM
          REDREC
          GETCHR
          GNSCHR
          STOCHR
          DECASC
          MMREAD
          OUTREC
          LUTBLS
          DGNTAB
          PICKUP
          P2NAM3
*T
*K, I8
*N, CONF2C, , , B
*K, I6
*P, F, GO2
          CONFIG
          GOCONF
          SCDKIO
          ERROR
          DCTOAS
          GETITM
          CALADR
          INCPTR
          GETFLE
          GO2
          PHASE2
          EQUIVA
          INSERT
          DELETE
          GETVAL
          CVTNUM
          GETNUM
          REDREC
          GETCHR
          GNSCHR
          STOCHR
          DECASC
          MMREAD
          OUTREC
          OUTLNN
          FTNMSK
          PRESET
          PICKUP
          P2NAM4
*T
*K, I8
*N, CONF2D, , , B
*K, I6
*P, F, GO3A
          CONFIG
          GOCONF
          SCDKIO
          ERROR
          DCTOAS
          GETITM
          CALADR
          INCPTR
          GETFLE
          GO3A
          PHASE3
          PACKAG
          INSPGM
          DELPGM
          OUTORD
          XTCORE
          INPBIN
          OUTBIN
          UNLOAD
          GETVAL
          CVTNUM
          GETNUM
          GETCHR
          GNSCHR
          STOCHR
          BINASC
          PICKUP
          PAGEJT
          PRNTLN
          PACKLN
          NEWHDR
          STAPGM
          STAPCK
*T
*K,I8
*N,CONF3A,,,B
*K,I6
*P,F,GO3B
          CONFIG
          GOCONF
          SCDKIO
          ERROR
          DCTOAS
          GETITM
          CALADR
          INCPTR
          GETFLE
          GO3B
          INPBIN
          GETVAL
          CVTNUM
          GETNUM
          GETCHR
          GNSCHR
          BINASC
          PICKUP
          PAGEJT
          PRNTLN
          PACKLN
          OUTBIN
          STAEND
   *T
   *K,I8
   *N,CONF3B,,,B
   *U
```

System Definitions and Skeletons Tape

The system definitions and skeletons magnetic tape contains the system definitions, SYSDAT skeleton and the system skeletons and terminates with a double end-of-file.

system definitions = 239 BCD records

SYSDAT skeleton = 1877 BCD records

system skeleton = 1638 binary records

```
**                              DEFINITION LIST
**
**
*SYSTEM HARDWARE DEVICES
**
**     CORE MEMORY. COMPONENT, INCLUDING THE INTERNAL INTERRUPT
*+CORE,
*      CORE STACKS                                  (8)        (4,8)
*      PRIORITY LEVELS                              (16)       (8,16)
*      SCHEDULER ENTRIES                            (25)       (10,100)
*      EXTRA VOLATILE LOCATIONS                     (0)        (0,100)
*      NON-SYSTEM CORE LOCATIONS                    (0)        (0,$6000)
*      SWAP TIME IN SECONDS                         (0)        (0,600)
*      INTERNAL INTERRUPT LEVEL                     (15)       (13,15)
*      CORE ALLOCATOR LEVEL                         (7)        (6,8)
*      ALLOCATABLE AREA LENGTHS (,,,,,,,,,,,)                  (1,$6000)
*      PRESETS (,,,,,,,,,,,,,,,,,,,,,,,,,,)                    (ANY)
**
**     1721/1722 PAPER TAPE READER
*+1721,
*      INTERRUPT LEVEL                              (10)       (8,12)
*      LOGICAL UNIT                                 (2)        (2,127)
*      ALTERNATE LOGICAL UNIT                       ()         (2,127)
**
**     1723/1724 PAPER TAPE PUNCH
*+1723,
*      INTERRUPT LEVEL                              (10)       (8,12)
*      LOGICAL UNIT                                 (3)        (2,127)
*      ALTERNATE LOGICAL UNIT                       ()         (2,127)
**
```

```
**      1711/1712 TELETYPE
*+1711,
*       INTERRUPT LEVEL                          (10)          (8,12)
*       LOGICAL UNIT                             (4)           (2,127)
*       ALTERNATE LOGICAL UNIT                   ( )           (2,127)
**
**      1713 TELETYPE WITH READER/PUNCH
*+1713,
*       INTERRUPT LEVEL                          (10)          (8,12)
*       LOGICAL UNIT                             (4)           (2,127)
*       ALTERNATE LOGICAL UNIT                   ( )           (2,127)
*       READER LOGICAL UNIT                      (2)           (2,127)
*       READER ALTERNATE LOGICAL UNIT            ( )           (2,127)
*       PUNCH LOGICAL UNIT                       (3)           (2,127)
*       PUNCH ALTERNATE LOGICAL UNIT             ( )           (2,127)
**
**      1731 MAG TAPE CONTROLLER WITH 601 TAPE UNITS
*+1731/601,
*       INTERRUPT LINE                           (15)          (2,15)
*       INTERRUPT LEVEL                          (14)          (12,15)
*       TAPE UNITS                               (2)           (1,8)
*       LOGICAL UNITS                            (6,7,,,,,,)   (2,127)
*       ALTERNATE LOGICAL UNITS                  (,,,,,,,)     (2,127)
*       EQUIPMENT NUMBER                         (7)           (0,15)
**
**      1732 MAG TAPE CONTROLLER WITH 608 or 609 TAPE UNITS
**1732/608-9,
*       INTERRUPT LINE                           (3)           (2,15)
*       INTERRUPT LEVEL                          (11)          (9,13)
*       TAPE UNITS                               (1)           (1,8)
*       LOGICAL UNITS                            (6,7,,,,,,)   (2,127)
*       ALTERNATE LOGICAL UNITS                  (,,,,,,,)     (2,127)
*       EQUIPMENT NUMBER                         (7)           (0,15)
*       BUFFERED I/O                             (NO)          (YES,NO)
*       CORE RESIDENT DRIVER                     (YES)         (YES,NO)
*       608 TAPE UNITS                           (NO)          (YES,NO)
**********************************************************************************************
**      THE FOLLOWING IS USED TO GENERATE 608 AND 609 UNITS.  IF UNITS 6 AND 7 ARE
**      SELECTED THE FOLLOWING WILL GENERATE A 608 ON 6 AND A 609 ON 7.
**      MIXED 608/609                            (YES)
**      608 UNITS                                (YES,NO)
**********************************************************************************************
*       MIXED 608/609                            (NO)          (YES,NO)
*       608 UNITS                                (,,,,,,,)     (YES,NO)
**********************************************************************************************
```

```
**      1738 DISK CONTROLLER WITH 853 or 854 DISK DRIVES
*+1738/853-4,                                                            0800
*       INTERRUPT LINE                        (4)         (2,15)           01
*       INTERRUPT LEVEL                       (9)         (8,10)           02
*       DISK DRIVES                           (1)         (1,2)            03
*       LOGICAL UNITS                         (8,)        (2,127)          04
*       ALTERNATE LOGICAL UNITS               (,)         (2,127)          06
*       EQUIPMENT NUMBER                      (3)         (0,15)           08
*       853 DISK DRIVES                       (YES)       (YES,NO)         09
**      THE FOLLOWING WILL GENERATE ONE 853 AND ONE 854.
**      MIXED 853/854                         (YES)
**      853 UNITS                             (YES,NO)
***********************************************************************************************
```

```
*        MIXED 853/854                               (NO)       (YES,NO)
*        853 UNITS                                   (,)        (YES,NO)
**
**       1728/430 CARD READER/PUNCH
*+1728/430,
*        COMMON INTERRUPT LINE                       (3)        (2,15)
*        INTERRUPT LEVEL                             (14)       (12,15)
*        LOGICAL UNIT                                (5)        (2,127)
*        ALTERNATE LOGICAL UNIT                      ( )        (2,127)
*        EQUIPMENT NUMBER                            (11)       (0,15)
**
**       1726/405 CARD READER
*+1726/405,
*        INTERRUPT LINE                              (5)        (2,15)
*        INTERRUPT LEVEL                             (8)        (8,15)
*        LOGICAL UNIT                                (5)        (2,127)
*        ALTERNATE LOGICAL UNIT                      ( )        (2,127)
*        EQUIPMENT NUMBER                            (4)        (0,15)
**
**       1729 CARD READER
*+1729,
*        INTERRUPT LEVEL                             (10)       (8,12)
*        LOGICAL UNIT                                (5)        (2,127)
*        ALTERNATE LOGICAL UNIT                      ( )        (2,127)
**
**       1729-2 CARD READER
*+1729-2,
*        INTERRUPT LINE                              (11)       (2,15)
*        INTERRUPT LEVEL                             (13)       (11,15)
*        LOGICAL UNIT                                (5)        (2,127)
*        ALTERNATE LOGICAL UNIT                      ( )        (2,127)
*        EQUIPMENT  NUMBER                           (12)       (0,15)
**
**       1740/501 LINE PRINTER
*+1740/501,
*        INTERRUPT LINE                              (12)       (2,15)
*        INTERRUPT LEVEL                             (10)       (8,12)
*        LOGICAL UNIT                                (9)        (2,127)
*        ALTERNATE LOGICAL UNIT                      ( )        (2,127)
*        FORTRAN LOGICAL UNIT                        ( )        (2,127)
*        ALTERNATE FORTRAN LOGICAL UNIT              ( )        (2,127)
*        EQUIPMENT NUMBER                            (15)       (0,15)
**
**       1742 LINE PRINTER
*+1742,
*        INTERRUPT LINE                              (12)       (2,15)
*        INTERRUPT LEVEL                             (10)       (8,12)
*        LOGICAL UNIT                                (9)        (2,127)
*        ALTERNATE LOGICAL UNIT                      ( )        (2,127)
*        FORTRAN LOGICAL UNIT                        ( )        (2,127)
*        ALTERNATE FORTRAN LOGICAL UNIT              ( )        (2,127)
*        EQUIPMENT NUMBER                            (15)       (0,15)
```

```
**      1573 LINE SYNC CLOCK
*+1573,
*       INTERRUPT LINE                          (2)         (2,15)
*       INTERRUPT LEVEL                         (13)        (11,15)
*       EQUIPMENT NUMBER                        (8)         (0,15)
*       CLOCK FREQUENCY IN CYCLES/SECOND        (60)        (50,7680)
*       MAXIMUM SCHEDULES/TIME PERIOD           (5)         (1,10)
**
**      DUMMY INPUT/OUTPUT DEVICE
*+DUMMY,
*       LOGICAL UNIT                            ( )         (2,127)
*       DUMMY LEVEL                             (10)        (8,10)
**
**      STANDARD LOGICAL UNITS
*+STANDARD UNITS,
*       INPUT UNIT                              (2)         (2,127)
*       OUTPUT UNIT                             (3)         (2,127)
*       LIST UNIT                               (9)         (2,127)
*       COMMENT INPUT UNIT                      (4)         (2,127)
*       COMMENT OUTPUT UNIT                     (4)         (2,127)
*       LIBRARY UNIT                            (8)         (2,127)
*       SCRATCH UNIT                            (8)         (2,127)
**
**      INSERT COMPONENT FOR SYSTEM DATA PROGRAM
*+INSERT,
*       INPUT FROM LOGICAL UNIT                 ( )         (2,127)
*       REGION                                  ( )     (PROCES,MISCEL)
**
**
**
*CORE RESIDENT FOREGROUND PROGRAMS
**
**      E006*2.1 MONITOR PACKAGE
*+MONITOR,
*       DISK WORD ADDRESSING                    (NO)        (YES,NO)
**
**      INSERT COMPONENT FOR CORE RESIDENT FOREGROUND PROGRAMS
*+INSERT,
*       INPUT FROM LOGICAL UNIT                 ( )         (2,127)
*       ORDINAL NAME                            ( )         (ANY)
**
**
**
*MASS RESIDENT FOREGROUND PROGRAMS
**
**      JOB PROCESSOR WITH LOADER, LIBRARY EDIT, BREAKPOINT, RECOVERY
*+JOB PROCESSOR,
*       MANUAL INTERRUPT PROCESSOR              (NO)        (YES,NO)
*       ON-LINE DEBUG PROGRAM                   (NO)        (YES,NO)
**
```

```
**      INSERT COMPONENT FOR MASS RESIDENT FOREGROUND PROGRAMS
*+INSERT,
*       INPUT FROM LOGICAL UNIT                  ()          (2,127)
*       ORDINAL NAME                             ()          (ANY)
**
**
**
*CORE RESIDENT FOREGROUND PROGRAMS
**
**      COMPONENT TO ANTICIPATE PROGRAMS THAT WILL BE ADDED IN THE FUTURE
*+PROGRAMS TO BE ADDED.
*       ESTIMATED NUMBER OF LOCATIONS            (0)         (0,$6000)
**
**      INSERT COMPONENT FOR CORE RESIDENT FOREGROUND PROGRAMS
*+INSERT,
*       INPUT FROM LOGICAL UNIT                  ()          (2,127)
**
**
**
*PROGRAM LIBRARY PROGRAMS
**
*+FTN RUNTIME LIBRARY,
*       ARITHMETIC FUNCTIONS                     (YES)       (YES,NO)
*       FORTRAN INPUT/OUTPUT                     (YES)       (YES,NO)
**
**      INSERT COMPONENT FOR PROGRAM LIBRARY PROGRAMS
*+INSERT,
*       INPUT FROM LOGICAL UNIT                  ()          (2,127)
**
**
**
*TERMINATE
```

The system skeleton contains the following system programs listed in the order in which they appear on the tape.

TRVEC
COMMON
NIPROC
MEPROC
NMONI
PARAME
ALVOL
OFVOL
NCMPRQ
NFNR
MAKQ
ALCORE
DRGORE
MINT
RW
ADEV
SCHEDU
NDISP
TMINT
DTMER
LOAD
BRANCH
LIDRIV
LCDRIV
LMDRIV
LLDRIV
SCAN
CHPU
ADJOVF
CONVRT
TABSCH
TABSTR
LSTOUT
LINK1
LINK2
COREXT
DPRADD
LOADER
NAMPRO
RBDBZS
ENTEXT
XFRPRO
HEXPRO
EOLPRO
ADRPRO
JOBENT
T7
T11
T3
JOBPRO
PROTEC
T5
JPLOAD

JPST
JPCHGE
ASCHEX
JBKILL
JPT13
T13
RESTOR
LIBEDT
UTILIB
PLINSN
FILE
GENLIB
BRKPTD
BIASCI
SIFT
RETJMP
JUMPTO
ENTER
ENTCOR
PRTREG
SETBRP
TERMIN
DMPCOR
MASDMP
RESUME
RCOVER
OUTSEL
DMPCOR
MASDMP
MIPRO
ODEBUG
DRVMAC
S13002
DRVMAC
S13OO3
DR1732
DR1732
PTREAD
PUNCDR
TELTYP
S13001
MASDRV
DR1732
DR1732
TAPCOR
TAPE
DISKWD
DISK
PUN415
DR1728
CR405

DR1729
CD1729
PRT40
PRINTR
XTRCOR
SPACE
Q8EXPN
Q8PRMS
Q8AB
IFALT
SIGN
FXFL
EXPPRG
SQRTF
LNUPRG
TANH
SINCOS
ARCTRG
FUCAT
FORTRA
Q8QINI
Q8QEND
Q8CMP
Q8RWBU
Q8ERRM
Q8DFIO
Q8QX
Q8QUNI
Q8FGET
Q8MAGT
TAPCON
IOCK
PSSTOP
Q8PAND
Q8EXP9
Q8EXP1
Q8IFRM
Q8FS
Q8TRAN

COSY Source Tape

The SYSCON COSY tape contains both FORTRAN and assembly language programs and terminates with an end-of-file mark.

The deck names for the FORTRAN programs are as follows:

VERIFY
BKCMVR
CALADR
CNVTNO
CONTRL
CONVRT
CORECK
CORECT
DCTOAS
DEFINE
GETCHR
GETITM
INCINS
INCPTR
INITAL
INITCM
INSURT
OPTCHK
PAMCHK
PAMCH2
PARCHK
PARTIT
RDSKEL
RNGCHK
SCNOPT
SCNREC
SEARCH
SPCPAR
SPECF1
SPECF2
STOCHR
SYSDAT
SYSINS
VALCHK
VALPRO
VARPRO
WRTMMR
PHASE2
CVTNUM
DECASC
DELETE
DGNTAB
EQUIVA
FTNLVL
FTNMSK

GETNUM
GETVAL
GNSCHR
HICORE
INSERT
INTREG
LUTBLS
MMREAD
MSKTBL
OUTLNN
PRESET
REDREC
SCHSTK
PHASE3
BINASC
DELPGM
INPBIN
INSPGM
NEWHDR
OUTORD
PACKAG
STAEND
STAPCK
STAPGM
XTCORE

The deck names for the assembly language programs are as follows:

COMMNT
CONFIG
ERROR
GETFLE
GOCONF
GO1A
GO1B
GO1C
GO1D
GO1E
GO1F
GO2
GO3A
GO3B
INPREC
INSINP
MESSGS
OUTBIN
OUTREC
PACKLN
PAGEJT
PICKUP
PRNTLN
P2NAM1

```
P2NAM2
P2NAM3
P2NAM4
SCDKIO
UNLOAD
SPACE
```

System Configurator Verification: The verification program is on the COSY source tape under the deck name VERIFY. Transfer the program to either paper tape, cards, or magnetic tape.

```
**            VERIFICATION DECK FOR SYSCON
**            SPECIFICATION LIST
**
*SYSTEM HARDWARE DEVICES
**
**   INVALID COMPONENT--USED TO VERIFY CONVERSE OPTION
*+1703,
**   1723/1724 PAPER TAPE PUNCH
*+1723,
**   1711/1712 TELETYPE
*+1711,
**   1738 DISK CONTROLLER WITH 853-4 DISK DRIVES
*+1738/853-4,
**
*CORE RESIDENT FOREGROUND PROGRAMS
**
**   E006*2.1 MONITOR PACKAGE
*+MONITOR,
**
*MASS RESIDENT FOREGROUND PROGRAMS
**
**   JOB PROCESSOR WITH LOADER, LIBRARY EDIT, BREAKPOINT, RECOVERY
*+JOB PROCESSOR,
**
**
*PROGRAM LIBRARY PROGRAMS
**
*+FTN RUNTIME LIBRARY,
**
*TERMINATE
```

4.4 ECO LEVELS

4.4.1 ECO LEVEL OF 1700 SERIAL 0

The following are the recommended ECO levels since the released version of MSOS has been tested on a system at these levels. As far as is known, however, it is not mandatory to be at this level.

Equipment	ECO Level
1704	A30
1705	A02
1703	A02
1708	A01
1709	A01
1713	A07
1731	A10
1738	A09
1742	A05
1740	A02
1716	A05
1706	A03
1721	A03
1723	A06
1729	A06
853	A05
601	A04
501	C11
430	A02
1729-2	A01

4.4.2 ECO LEVELS OF PRODUCT SET AND DRIVERS

1728-430	Reader-Punch	A01
1729-2	Card Reader 2.1	A01

Mass Storage FORTRAN 2.0 is the same level as MSOS 2.1.

4.5 INSTALLATION VERIFICATION PROGRAMS

4.5.1 OPERATING SYSTEM AND MACRO ASSEMBLER

To verify that the operating system and the Macro Assembler are installed:

1. Ready the system for operation
2. Press: AUTOLOAD on the 1738 disk controller

3. Set the STEP/RUN switch to RUN

4. Message: TIMER RJ

   PP

5. Set the PROGRAM PROTECT switch

6. Type: *

   Press: CARRIAGE RETURN

   Press: MANUAL INTERRUPT on teletypewriter

   Message: MI

7. Type: *P

   Press: CARRIAGE RETURN

   Message: J

8. Ready the Macro Assembler verification program in the card reader.

9. Type: *V, 11

   Press: CARRIAGE RETURN

   Message: MACRO ASSEMBLER IS INSTALLED

   J

4.5.2 COSY 1.0

To verify that COSY is installed:

1. Ready the system for operation

2. Press: AUTOLOAD on the 1738 disk controller

3. Set the STEP/RUN switch to RUN

4. Message: TIMER RJ

   PP

5. Set the PROGRAM PROTECT switch

6. Type: *

   Press: CARRIAGE RETURN

   Press: MANUAL INTERRUPT on the teletypewriter

   Message: MI

7. Type: *P

   Press: CARRIAGE RETURN

   Message: J

8. COSY verification deck can be used with magnetic tape only. Mount and ready at loadpoint two magnetic tapes on LUN's 6 and 7. Ready the released COSY verification deck in the card reader.

9. Type: *V, 11

   Press: CARRIAGE RETURN

   Message: COSY IS INSTALLED

   J

4.5.3 MASS STORAGE FORTRAN 2.0A AND 2.0B

To verify that FORTRAN is installed:

1. Ready the system for operation
2. Press: AUTOLOAD on the 1738 disk controller
3. Set the STEP/RUN switch to RUN
4. Message: TIMER RJ

   PP
5. Set the PROGRAM PROTECT switch
6. Type: *

   Press: CARRIAGE RETURN

   Press MANUAL INTERRUPT on the teletypewriter

   Message: MI
7. Type: *P

   Press: CARRIAGE RETURN

   Message: J
8. Ready the released FORTRAN verification deck in the card reader
9. Type: *V, 11

   Press: CARRIAGE RETURN

   Message: OPTIONS
10. Type: LX

    Press: CARRIAGE RETURN

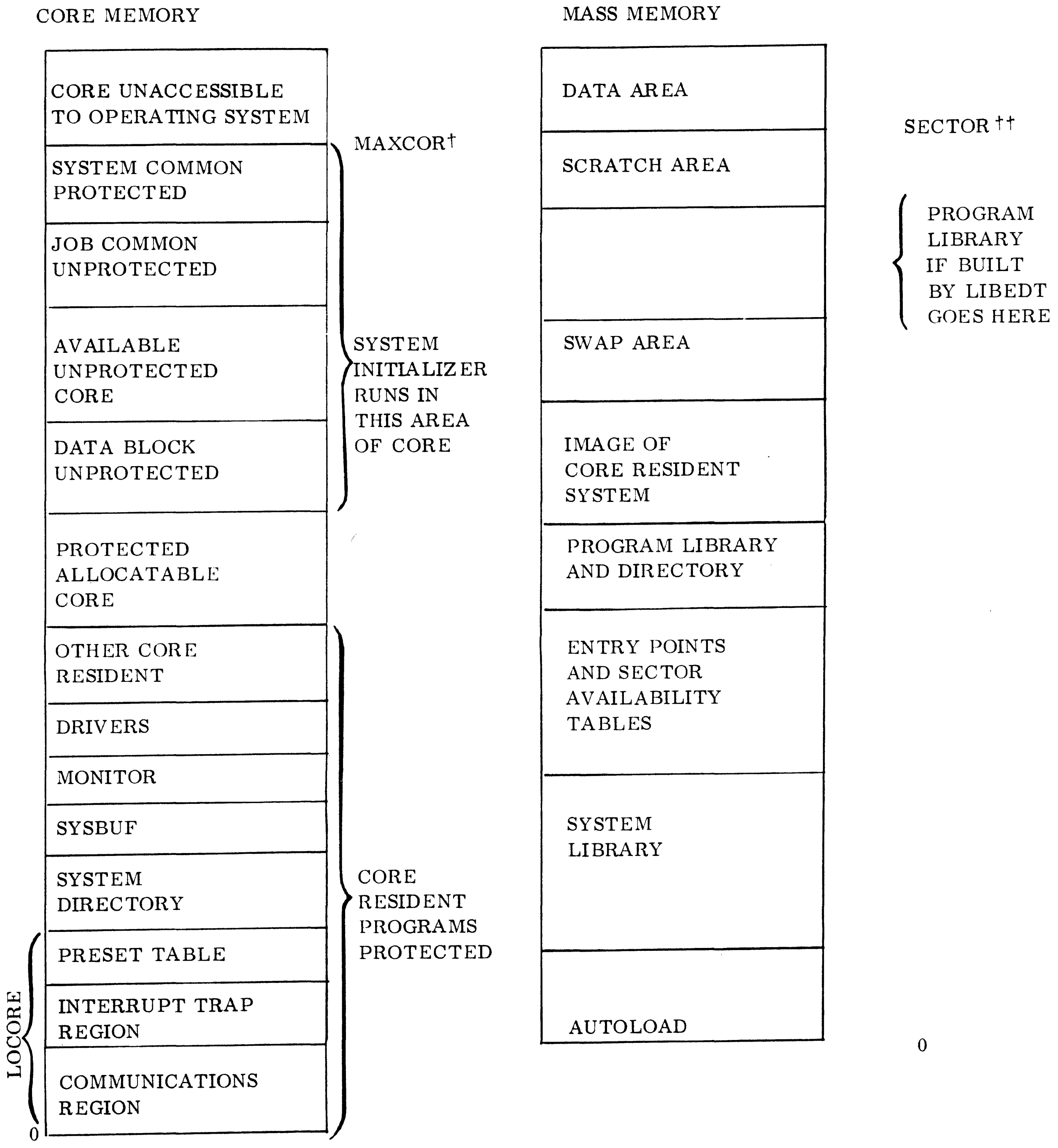
    Message: FORTRAN IS INSTALLED

    J

PART III

INSTALLATION-RELATED INFORMATION

CUSTOMIZATION 1


CORE MEMORY
CORE UNACCESSIBLE TO OPERATING SYSTEM
MAXCOR†
SYSTEM COMMON PROTECTED
JOB COMMON UNPROTECTED
AVAILABLE UNPROTECTED CORE
SYSTEM INITIALIZER RUNS IN THIS AREA OF CORE
DATA BLOCK UNPROTECTED
PROTECTED ALLOCATABLE CORE
OTHER CORE RESIDENT
DRIVERS
MONITOR
SYSBUF
SYSTEM DIRECTORY
CORE RESIDENT PROGRAMS PROTECTED
PRESET TABLE
LOCORE
INTERRUPT TRAP REGION
COMMUNICATIONS REGION
0
MASS MEMORY
DATA AREA
SCRATCH AREA
SECTOR ††
PROGRAM LIBRARY IF BUILT BY LIBEDT GOES HERE
SWAP AREA
IMAGE OF CORE RESIDENT SYSTEM
PROGRAM LIBRARY AND DIRECTORY
ENTRY POINTS AND SECTOR AVAILABILITY TABLES
SYSTEM LIBRARY
AUTOLOAD
0

† Parameter specified during system initialization determines this area. Section 1.1.1.

†† Parameter specified during system initialization specifies limit of available core.

1.1 LOCORE

The LOCORE program consists of data to be loaded into the communications region, interrupt traps, and preset table. During system initialization, the LOCORE program must be the first *L program loaded after the *Y, *YM system directory entries.

NOTE

> If any core-resident system directory entry (*Y) is included, the ordinal must be two or greater, since the first program loaded was LOCORE. LOCORE cannot be a system directory entry.

Part One of the LOCORE program corresponds to the communications region.

Part Two is the interrupt trap region from location 100_{16} to the maximum interrupt trap region used (which could be up to a maximum of $13F_{16}$).

Part Three of the LOCORE program is the table of presets specifying the name and location of entry points to any protected routines which are also available to unprotected programs.

Part Four is designated for use by the assembler or FORTRAN compiler and includes the maximum sector number of the scratch mass storage device.

The following modifications must be made by the system programmer for a specific system.

1.1.1 EQUIVALENCES

MAXCOR is the highest core memory address in hexadecimal available to the system. Core locations above MAXCOR are not affected by normal system operation and may be used for upper core routines, core dumps, etc. This parameter is derived by setting MAXCOR with a *S, MAXCOR, xxxx parameter during system initialization.

1.1.2 COMMUNICATIONS REGION

If required, communications region information can be inserted in the area from location 47_{16} through $B2_{16}$. These entries may be either numeric or the symbolic address of an entry point in another program. In the latter case, the symbolic address must also be declared as an external (EXT). Labels can be attached to these entries and, if declared as entry points (ENT), they can be referenced by other programs. Unused entries should be set to zero.

In the example below, the sequence of code replaces the block:

```
    BZS ($B2-$47+1)
```

When the program with entry points SNAPE and SNAPI is loaded, the initializer loads the addresses of SNAPE and SNAPI. Also, it stores a special table starting at location 52_{16} which may be referenced by the entry point name MTAB.

```
LABEL        OP       ADDRESS

             BZS      ($50-$47+1)
             ADC      SNAPE
             ADC      SNAPI
MTAB         NUM      $F
             NUM      $F0
             NUM      $F00
             NUM      $F000
             BZS      (B2-*+1)
             EXT      SNAPE, SNAPI
             EXT      MTAB
```

1.1.3 INTERRUPT TRAP REGION

The interrupt trap region extends from location 100_{16} to $13F_{16}$ in LOCORE. A four-location trap is necessary for each of the 16 interrupt lines which are used (Section 1.2.3, Interrupt Mask Table). For example, the LINE0 trap area contains four words beginning at word 100_{16}. LINE1 trap area then begins at word 104_{16}. The form of the four-word trap is:

```
WORD      LABEL        OP       ADDRESS

 1        LINEno.      NUM      0
 2                     RTJ-     ($FE)
 3                     NUM      level
 4                     ADC      interrupt response routine
```

Explanation of Each Location

Word 1

The hardware stores the state of overflow indicator in bit 15 and also stores the P register contents in bits 14-00. The P register contains the address of the next instruction to be executed when the program is later re-entered.

Word 2

The second word in the interrupt trap is normally used to pass control to the common interrupt handler which will:

1. Store the contents of the A, Q, P, and I registers and the current priority level (PRVL).
2. Establish priority of the program being entered through the third word of the trap.
3. Jump to the interrupt response routine through the fourth word of the trap.

Usually all interrupt lines (except for line 0) use the common interrupt handler whose address is in location FE_{16}.

Any special interrupt handler routines may be used to avoid the overhead required to go through the common interrupt handler. Include the address of the special interrupt handler routines in the communications region between locations 47_{16} and $B2_{16}$ and declare this address as external. The special interrupt handler must preserve the A, Q, P, and I registers and the overflow indicator and return control (with interrupts enabled) to the interrupted program after processing the interrupt. Save priority levels (PRVL) if the response routine runs with interrupts enabled.

Word 3

In word 3 is the priority level of the program which will process the interrupts on the specified line. When assigning priorities:

1. The number in word 3 must correspond with the interrupt mask table entry in MASKT of the SYSBUF or the TABLES program.
2. Priority levels assigned to peripheral devices cannot also be assigned to FORTRAN programs.
3. Because of timing problems, use caution when assigning priorities to devices which are subject to losing data. High priorities should be assigned to these devices, such as the 1729-2 Card Reader and unbuffered magnetic tape devices.
4. Interrupt lines for I/O drivers must be assigned the same priority level as that specified in the PHYSTB. That is, the initiator (CP in the appropriate PHYSTB) and the continuator (priority level PR in the appropriate interrupt trap entry) must be the same priority level.

Word 4

This is the address of the interrupt response routine which is the program which processes the interrupt. Each interrupt response routine name must be declared as an external in LOCORE.

External Interrupt Processor (EPROC)

EPROC is a generalized External Interrupt Processor. To use EPROC:

1. Declare it as external in LOCORE.
2. Device must return bit 2 as interrupt status upon a status request.
3. Add the SECPRO table to the SYSBUF program.

SECPRO is a 16-word table which is required only if EPROC is in use. It contains one word for each interrupt line. When EPROC cannot determine which device on a particular line caused an interrupt (indicated by bit 2 of device status), EPROC transfers control to the corresponding secondary processor for that line. SECPRO may contain up to 16 secondary processor addresses. Each location may refer to an entry point of a secondary interrupt processor. The first location of the table is declared entry point SECPRO. (Section 1.2.2, LOG1A Table and EPROC.) Limitations for using EPROC are as follows.

> Using EPROC instead of separate response routines for each line increases the interrupt processing time.
>
> Using any of the following special devices requires separate interrupt response routines:
>
> > 1573 Line Synchronized Timing Generator.
> >
> > Devices which do not give the interrupt status in bit 2 of the A register while a reply is being made to a status command.

Individual Interrupt Response Routines

Use the following rules when developing individual interrupt response routines.

1. If several devices are driven on the same interrupt line, the interrupt response routine must examine the status of each device to determine which one interrupted.
2. All interrupt response routines for drivers must branch to the driver's continuator entry with the address of the PHYSTB of the interrupting logical unit in the Q register.
3. Interrupt response routines usually reside in the SYSBUF program, except for EPROC.
4. Declare the address of each interrupt response routine as an external in LOCORE.

1.1.4 TABLE OF PRESET ENTRY POINTS

Definition

The preset table is a list of entry points of all programs in protected core, as well as all core-resident subprograms which can be used by jobs running in unprotected core.

Format

This is an example of a preset table entry. If the name of the entry point to the routine is NAME, the following code is required to add NAME to the preset table. The first entry must be for JPRETN.

```
LABEL        OP           ADDRESS        COMMENTS

             ALF          3,NAME
ALF          ADC          NAME
             EXT          NAME
```

The EQU for the table length must follow the last entry.

```
             EQU          LPRSET         (*APRSET) FOR THE LAST ENTRY
```

Rules

Use caution in constructing the preset table.

> The preset table must contain only references to subprograms which cannot destroy the integrity of the protected system.
>
> Subprograms which are referenced in the preset table must be re-entrant if they are also to be used by protected programs. They must have an IIN instruction immediately following each entry point. However, they do not need to be re-entrant if they are not to be used by protected programs.

Location

The preset table begins immediately following the interrupt trap region. The table starts at location 140_{16} if 16 interrupt lines are assigned. The table length is saved at location $F1_{16}$. The table starting address is saved at location $F2_{16}$.

1.1.5 MAXIMUM SCRATCH SECTOR NUMBER (MAXSEC)

Following the preset table is an area reserved for the use of the compiler or the assembler. The maximum sector number available on the scratch mass memory device (MAXSEC) is included in this area. MAXSEC is an initialization time parameter. This parameter is derived by setting SECTOR with an *S, SECTOR, xxxx parameter during system initialization. If part of mass storage is to be reserved for data storage not available to the system, MAXSEC is set to the maximum minus the amount reserved for data.

The area is defined as follows.

```
LABEL        OP           ADDRESS        COMMENTS

             ENT          MAXSEC
             BZS          (3)
             NUM          0              MSB OF MAX SECTOR
```

LABEL	OP	ADDRESS	COMMENTS
MAXSEC	NUM	SECTOR	LSB of MAX SECTOR
	BZS	(2)	

1.2 SYSBUF

The system and buffer tables program includes the following.

For use by the operating monitor:

Logical unit tables

Interrupt, scheduler, and timer stacks

Volatile storage for re-entrant routines

Diagnostic timer table

Routines and tables required by drivers:

Interrupt response routines

Physical device tables

Output message buffering package

Special routines:

Special error message routines

Dummy driver and device table

Overlay subroutine

Idle loop routine

1.2.1 EQUIVALENCES (EQU)

Set up SYSBUF equivalences (EQU) as required. Following is a list of EQU's.

EQU	SIGNIFICANCE
NUMPRI	Defines the total number of priority levels used by the operating system
NINTLV	Defines the number of priority levels used by interrupts
NFTNLV	Defines the number of priority levels using the re-entrant FORTRAN library
NEDLVL	Defines the number of priority levels using the re-entrant encode/decode package

EQU	SIGNIFICANCE
NSR	Defines the maximum number of programs that the timer program may schedule when a single timer interrupt occurs. Delete if the timer is not used
TIMACK	Defines the 1573 timer interrupt acknowledge code. Delete if the timer is not used
TIMCPS	Defines the 1573 timer frequency (Hz). Delete if timer is not used
TODLVL	Defines time-of-day routine request code and priority level. Delete if no time-of-day routine is used

Equivalences are included at appropriate locations in the LOG1A table to identify system logical units.

EQU	SIGNIFICANCE
STDINP	Logical unit number of the standard input device, e.g., paper tape or card reader or magnetic tape
BINOUT	Logical unit number of the standard binary output device, e.g., paper tape or card punch or magnetic tape
LSTOUT	Logical unit number of the standard print output device, e.g., teletypewriter or line printer
INPCOM	Logical unit number of the standard input comment device, e.g., teletypewriter
OUTCOM	Logical unit number of the standard output comment device, e.g., teletypewriter
LBUNIT	Logical unit number of the library mass storage device, e.g., disk or drum
SCRTCH	Logical unit number of the scratch mass storage device, e.g., disk or drum
DUMALT	Logical unit number of the dummy device driver

1.2.2 LOGICAL UNIT TABLES

The logical unit tables contain information for all logical units.

LOG1A contains the addresses of physical equipment tables for each logical unit. The order of these addresses reflects the logical assignment of the physical devices in LOG1A.

LOG1 contains the operational flags and alternate logical unit assignments.

LOG2 contains the top of request thread for each logical unit.

Each logical unit number has a corresponding entry in these tables. When using EPROC, the logical units are grouped according to which interrupt line they use. For example, devices which interrupt on line 1 are grouped after the L1 EQU in LOG1A. This construction is the same for all logical unit tables. Those devices which interrupt on line 2 are grouped after L2.

These logical unit tables are arranged to be parallel in structure and are indexed by logical unit number. The following apply to all logical unit tables.

Word 0 is always the maximum logical unit number or the table length-1

Word 1 is always the core allocator (the SPACE driver)

Other logical unit numbers are assigned according to the order in which the LOG1A is established.

LOG1A

LOG1A contains the address constants of the PHSTB's. Each word in LOG1A contains the address of the first word of that logical unit's PHYSTB. Since there is a PHYSTB for each device, the next LOG1A word contains the address of the first word of the next PHYSTB.

When using EPROC, all physical devices are grouped according to the interrupt lines which they use. Therefore, all physical devices interrupting on line one are grouped after entry L1 in the LOG1A. But the logical unit numbers assigned to each of these units are determined by the order in which each of these are arranged within the entry.

When using a user-supplied interrupt response routine, instead of EPROC, the tags (L1, L2, etc.) are irrelevant; but the devices must still be in logical unit order.

LOG1A FORMAT

LOG1A	+0	Largest legal logical unit number
	+1	Address of core allocator
	+2	Address of PHYSTB slot corresponding to this logical unit
	+3	
	+n	Address of PHYSTB slot corresponding to this logical unit

LOG1A Table and EPROC: If the LOG1A table is to be used with the external interrupt processor (EPROC), the following additional construction is necessary.

1. Group the devices by interrupt line number. This fixes the logical unit assignment.
2. Insert fifteen EQU statements of the form EQU Lx(*) (where x is a number from 1 to 15) in LOG1A. These EQU's are then used to identify the line number for the groups of devices. For example, EQU L1(*) precedes the device table addresses for the devices which interrupt on line 1. These are followed by EQU L2(*) and the device table addresses for the devices which interrupt on line 2, etc. To illustrate:

```
LABEL       OP       ADDRESS

LOG1A       NUM      NUMLU
            ADC      CORE        LU 1
            EQU      L1(*)
            ADC      PPTRDR      LU 2  }
            ADC      PPTPCH      LU 3  }  interrupt line 1 devices
            ADC      TELPTR      LU 4  }
            ADC      CARD29      LU 5  }
            EQU      L2(*)
            EQU      L3(*)
            EQU      L4(*)
            ADC      DISK0       LU 6  }
                                       }  interrupt line 4 devices
            ADC      DISK1       LU 7  }
            EQU      L5(*)
```

3. Construct the SECPRO table (see SECPRO, Section III.1.1.3):

```
SECPRO      NUM      $7FFF, $7FFF, $7FFF, $7FFF, $7FFF, $7FFF
            NUM      $7FFF, $7FFF, $7FFF, $7FFF, $7FFF, $7FFF
```

Normally, all entries are left empty, i.e., $7FFF. The address of a special interrupt response routine may be included in the entry for its line, but it is more efficient to put this address in the fourth word of the interrupt trap location instead of using EPROC or SECPRO.

If EPROC is not used, the logical unit assignment numbers do not need to be equated to the interrupt lines.

To use a logical unit order which differs from the interrupt line order to which the peripheral devices are connected, use separate interrupt response routines.

LOG1

LOG1 is the alternate device table. Unless an alternate device or shared LUN is to be specified, entries in this table are initially set to 0. If an alternate device is to be assigned, set bits 9-0 to the alternate logical unit number.

If a device fails, the driver calls the alternate device handler with the logical unit of the failed device. The alternate device handler checks the LOG1 entry for this logical unit and if a nonzero alternate logical unit is found, the request is rethreaded on the alternate LUN and the driver for the alternate is scheduled to process the request. A message is also typed. If the alternate logical unit is out of service or has failed, the request is passed to the alternate of the alternate, etc. A message also appears. If no operational alternate exists, a request for operator intervention is made.

If two or more logical units share the same device table, set bit 14 of the corresponding LOG1 entry to 1.

The order of entries in the LOG1 is identical to that of the LOG1A.

LOG1 FORMAT

LOG1	Largest legal logical unit number						
	15	14	13	12	11	10	9 Alternate logical unit number

Bit	Significance	
15	0	Normal logical unit
	1	Buffer output logical unit
14	0	Logical unit does not share device with another logical unit
	1	Logical unit shares a device with another logical unit
13	0	Logical unit is operative
	1	Logical unit is out of service. Alternate, if any, is in use
12		Reserved
11	0	No operation
	1	If need to, restore logical unit on completion of buffer output request
10		Reserved
9-0		Alternate logical unit number should be set to the hexidecimal equivalent of the logical unit number

LOG2

LOG2 contains the top of thread for each logical unit. The order of entries in LOG2 is identical to the order of entries in LOG1.

Entries are initially set to $FFFF_{16}$.

LOG2 FORMAT

LOG2	Largest legal logical unit number
	Top of thread for a logical unit number
	Top of thread for a logical unit number

1.2.3 INTERRUPT MASK TABLE

MASKT is a table of M register interrupt line mask words which are arranged in the software priority level order. Only the monitor may change the M register. It uses the MASKT to set the M register according to the current priority level.

Standard MASKT

Most of the operating system programs have been assigned to the standard priority levels shown in the following table.

Level	System Program
-1	idle loop
0	job processor execution
1	job processor I/O completion
2	hang loop while a SWAP is in effect
3	manual interrupt processor
4	process programs
5	process programs
6	process programs
7	core allocator
8	EOP for 1728 and 1729-2 card readers

Level	System Program
9	disk, drum, and output message buffering package
10	printer, paper tape punch, and paper tape reader
11	magnetic tape drivers
12	card reader, unbuffered magnetic tape
13	timer interrupt and event counters; card reader
14	
15	memory parity/protect fault routine

Construction and/or Modification of MASKT

The first step in constructing the MASK table is the assignment of software priorities. Follow these general concepts when developing the table.

1. Bits 0 through 15 of the M register correspond to interrupt lines 0 through 15. If, for example, bit 1 in the M register is set to zero, interrupts on interrupt line 1, the corresponding interrupt line, are locked out and are not processed until bit 1 in the M register is changed to a one.
2. Only the monitor can change the M register. It uses the MASKT to set the M register according to the current priority level.
3. Level -1 is used for the idle loop which must not include any monitor requests.
4. Each interrupt line normally has a 1 bit in the interrupt line position for all levels below the priority level associated with that line.
5. 0 bits must be placed in the interrupt lines position for all the priority levels equal to and above the priority level associated with the line.
6. Unused interrupt lines should be set to zero for each table entry.
7. More than one line can be associated with the same priority and can have the same mask.

Sample MASKT

PRIORITY LEVEL	INTERRUPT LINE 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	MASK_{16}
-1	0	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	043F
0	0	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	043F
1	0	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	043F
2	0	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	043F
3	0	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	043F
4	0	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	043F
5	0	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	043F
6	0	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	043F
7	0	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	043F
8	0	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	043F
9	0	0	0	0	0	1	0	0	0	0	1	0	1	1	1	1	042F
10	0	0	0	0	0	1	0	0	0	0	0	0	1	1	0	1	040D
11	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	1	0405
12	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0005
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0001
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0001
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000

Assembly language coding for this sample MASKT is:

```
LABEL        OP        ADDRESS        COMMENTS

             NUM       $43F           PRIORITY LEVEL -1
MASKT        NUM       $43F           PRIORITY LEVEL 00
             NUM       $43F           PRIORITY LEVEL 01
             NUM       $43F           PRIORITY LEVEL 02
             NUM       $43F           PRIORITY LEVEL 03
             NUM       $43F           PRIORITY LEVEL 04
             NUM       $43F           PRIORITY LEVEL 05
             NUM       $43F           PRIORITY LEVEL 06
             NUM       $43F           PRIORITY LEVEL 07
```

LABEL	OP	ADDRESS	COMMENTS
	NUM	$43F	PRIORITY LEVEL 08
	NUM	$42F	PRIORITY LEVEL 09
	NUM	$40D	PRIORITY LEVEL 10
	NUM	$405	PRIORITY LEVEL 11
	NUM	$05	PRIORITY LEVEL 12
	NUM	$01	PRIORITY LEVEL 13
	NUM	$01	PRIORITY LEVEL 14
	NUM	$00	PRIORITY LEVEL 15

1.2.4 VOLATILE STORAGE (VOLBLK)

Definition

VOLBLK is the volatile storage area which is primarily reserved for the allocation of small blocks of data storage for routines which are re-entrant (may operate at more than one level at the same time).

Allocation

Reserve enough volatile storage for each priority level to accommodate the maximum amount of volatile storage which could be requested at any one time because the system cannot recover from an overflow of volatile storage (i.e., requesting more storage than is available).

To compute allocation of volatile storage:

1. Allow 16 locations for each priority level making monitor requests. Eight of these locations are used for each request. The other eight locations may be used if the request processor itself makes a monitor request, such as the read/write request processor making a scheduler call for a driver.
2. Allow 49 locations (34 for locations $C5-$E5 and 15 for FLIST entry point addresses) for each priority level using the re-entrant FORTRAN library to allow the FORTRAN communications area and library subroutine entries to be saved.
3. Allow 56 locations for each priority level using the encode/decode package which is non-standard. The standard release equates this to zero.

The following code defines volatile storage (see SYSBUF equivalences in Section 1.2.1).

LABEL	OP	ADDRESS
VOLBLK	BSS	VOLBLK(16*NUMPRI+49*NFTNLV+56*NEDLVL+1)

1.2.5 INTERRUPT STACK AREA (INTSTK)

INTSTK is the block of storage which is set aside for saving the status of interrupted programs. The common interrupt handler stores the Q, A, I, and P registers and also the overflow indicator and the priority level of the interrupted program in this area. Five words are necessary for each entry. The stack is of the last-in, first-out type of stack on a priority basis.

The format of an entry is as follows.

	Word		
INTSTK	+0	Q register	Interrupted program running at priority level n
	1	A register	
	2	I register	
	3	Overflow (bit 15), P register	
	4	Priority level (=n)	
	5	Q register	Interrupted program running at priority level m
	6	A register	
	7	I register	
	8	Overflow (bit 15), P register	
	9	Priority level (=m)	Level m< n

The following code defines the interrupt stack.

```
LABEL     OP        ADDRESS

INTSTK    BSS       INTSTK(5 * NUM PRI)
```

1.2.6 SCHEDULER STACK (SCHSTK)

A program requests the operation of another program by making a scheduler (SCHDLE) request. The timer routine can also make a SCHDLE request after a given interval of time has elapsed. These requests are threaded together on the scheduler thread.

The scheduler stack (SCHSTK) is a series of four-word entries.

Words one and two contain the scheduler call parameters (priority level and address of program scheduled).

Word three contains the thread to the next lower priority entry.

Word four contains the value of the Q register which is being passed to the requested program as a parameter.

The total number of entries required is equal to the sum of the number of scheduler requests and timer requests which can be in the stack at one time. The user may change the size of this stack. Approximately 15 entries are sufficient for a small system.

Sample SCHSTK

```
LABEL     OP        ADDRESS        COMMENTS

SCHSTK    ADC       0,0,*+2,0      LEVEL, COMPLETION ADDR., THREAD, Q REG.
          ADC       0,0,*+2,0
                    .
                    .
                    .
                    .
          NUM       0,0,-0,0       LAST ENTRY
          EQU       SCHLNG
                    (*-SCHSTK)
```

1.2.7 ALLOCATABLE CORE (AVCORE)

EQU AVCOREnnnn is an entry in the SPACE program which defines the total size of the allocatable core area. CALTHD is the address of the location which contains the size of the first block which is initially all of allocatable core. Following this address is the address of the first piece (top of core allocator's thread) which is the beginning of the allocatable area.

No modification is necessary to the following code.

```
LABEL     OP        ADDRESS        COMMENTS

CALTHD    ADC       AVCORE         NO. OF WORDS
          ADC       AREAC          ADDRESS
```

LVLSTR is the table of starting addresses for the allocatable core area available to each priority level. The upper bound for protected allocatable area is the same for all levels — the start of unprotected core. To prevent low priority programs from tying up all of the allocatable area, it is common to restrict the amount available to them while making the entire allocatable area available to the high-priority programs. Thus, a higher address usually appears for the low-priority programs.

Core swapping occurs at the following times.

A request for space is made at a request priority level greater than two

No unprotected I/O is in progress

A fixed interval of time has expired since the last swap

There is insufficient space available to that priority level in the allocatable area

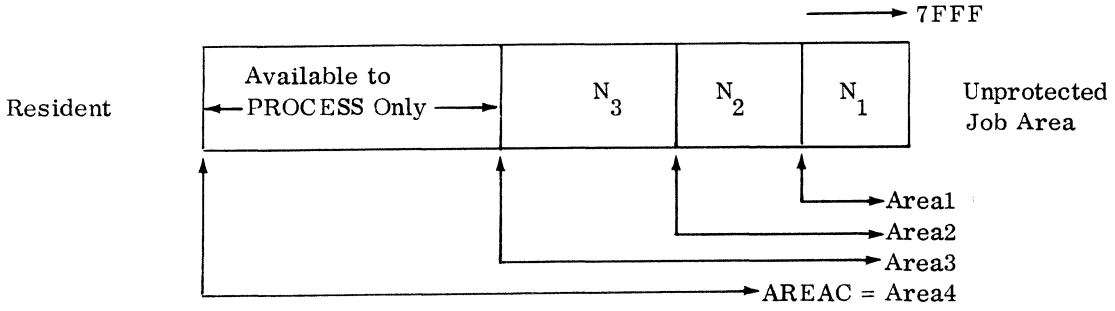
Version 2.1 of the operating system automatically causes a core swap whenever job processing is terminated. This causes the job area (unprotected allocatable core) to be protected and made available to protected mass memory resident programs. The swapped condition continues until job processing is requested again by the operator.

Example of LVLSTR:

```
LABEL      OP          ADDRESS          COMMENTS

           EXT         AREA1,AREA2,
                       AREA3,AREA4
LVLSTR     ADC         AREAC            0   REQUEST PRIORITY
           ADC         AREA1            1          LEVELS
           ADC         AREA2            2
           ADC         AREA3            3
           ADC         AREA4            4
           ADC         AREAC            5
           ADC         AREAC            6
                         .
                         .              7-15
                         .
                         .
           ADC         LEND
```

AREA1, AREA2, AREA3, and AREA4 are entry point names in the SPACE program used to divide the allocatable area, as shown in the diagram below (also refer to the SPACE program Section 1.3). Request priority levels 1, 2, and 3 include sufficient area for the job processor modules. The memory map for the LVLSTR table above is:



The entire allocatable core area (AREAC) must be available at request priority zero (RP equal to 0) so that the system may get started as initialized (job processor initiated with RP equal to 0 in system directory).

To make certain that the individual modules of the job processor can obtain sufficient allocatable core at all times, use the LIBEDT *S statements. Set the request priority for their system directory entries as follows. (This operation is done after the operating system is built and is functional. The *S statements should be entered only once after the system is built, since the mass memory image of the system directory is actually updated by the *S.)

*YM Entry Name	*YM Ordinal (Typical)	Request Priority (RP)
LOADSD	1	0
JOBENT	2	1
JOBPRO	3	2
JPLOAD	4	3
JPST	5	3
JPCHGE	6	3
JBKILL	7	3
JPT13	8	3
RCOVER	9	4
LIBEDT	10	2
MOD1	11	3
MOD2	12	3
MOD3	13	3
MOD4	14	3
RESTOR	15	4
ODEBUG	16	5
RCOVER	17	2
BRKPT	18	0

1.2.8 SPECIAL ROUTINES

IDLE

IDLE is the program which runs at level -1 when no other programs are running. This routine may be modified by the user. A counter may be included to compute the percentage of time spent at this level to provide a measure of the amount of idle time available in the main frame.

DUMMY

DUMMY is the dummy device driver. It is used with the dummy device table and is assigned a logical unit like a normal device. Read or write requests which address this logical unit cause the dummy driver to be initiated, and the completion address in the request is scheduled with error indication. This allows the dummy device to be set up as the alternate for devices where it would not be acceptable to hang up the request waiting for operator action in response to the alternate device handler request for input. This routine requires no modification.

FMASK, FLIST

FMASK and FLIST contain data for the re-entrant FORTRAN dispatcher and scheduler, RDISP. If the re-entrant FORTRAN library package and RDISP are used, FMASK and FLIST may require modification; if RDISP is not used, FMASK and FLIST may be removed. FMASK is a location which indicates the software priority levels which require the saving of the temporary area used by the FORTRAN routines. These levels must not also be assigned to interrupt lines since the interrupt handler does not save the FORTRAN data. A bit is set to 1 in FMASK in the bit position corresponding to each level using FORTRAN. If too many levels are allowed to run FORTRAN programs, the overhead for the low-priority programs may be unnecessarily high. For example, the following allows FORTRAN at levels 4, 5, and 6.

LABEL	OP	ADDRESS
FMASK	NUM	$0070

Levels 0 and 1 are reserved for unprotected programs and do not interrupt the priority levels using FORTRAN. Therefore, the mask is not set for levels 0 or 1.

FLIST Table

FLIST is the table of entry point locations in the FORTRAN library which must be saved in order to allow re-entrant use of the library. The symbolic names must also be declared as externals (EXT) and must appear as entry names (ENT) in the library subroutines.

CHRSFG

CHRSFG is a switch that indicates whether or not the on-line debug package (ODP) is running. When the debug package is running, CHRSFG is not zero.

Q8STP

Q8STP provides a branch to the dispatcher for FORTRAN object programs. It cannot be used by protected mass memory resident programs as a substitute for CALL RELESE main. The entry point name Q8STP is that generated by the compiler as an exit at the end of a compiled program.

NSCHED

NSCHED contains the maximum number of programs which may be scheduled per timer interrupt.

1.2.9 SPECIAL TABLES

Diagnostic Timer Table (DGNTAB)

DGNTAB is a table which consists of the PHYSTB addresses for all the devices to be supervised by the diagnostic timer program. Software buffer driver PHYSTB's may also be included in the table. The end of the table is indicated by a negative address, i.e., bit 15 = 1. Note that the first word in the table is not the table size.

To add a driver place an entry in the diagnostic table. Each entry is a pointer to the physical device table for that device.

Example:

```
LABEL     OP        ADDRESS         COMMENTS

          ENT       DGNTAB          DIAGNOSTIC TIMER TABLE
*
DGNTAB    ADC       CORE             1  CORE ALLOCATOR
          ADC       PPTRDR           2  1721 PAPER TAPE READER
          ADC       PPTPCH           3  1723 PAPER TAPE PUNCH
          ADC       TELPTR           4  1711 TELETYPE
          ADC       CARD29           5  1729 CARD READER
          ADC       TPPDR1           6  601 MAG. TAPE, UNIT0
          ADC       TPPDR2           7  601 MAG. TAPE, UNIT1
          ADC       DISK0            8  853 DISK
          ADC       LP1742           9  1742 LINE PRINTER
          ADC       CD1728          11  1728 CARD READER
          NUM       $FFFF               END OF TABLE
```

Alternate Device Handler (ALTERR)

ALTERR is the buffer table for the alternate device handler. It is used to save the error word (Q register) passed by a driver to the alternate device handler. Location ALTERR contains the table size, followed by a block of zeros of this size. The size should be set to the maximum number of simultaneous device failures possible. For most systems this equals the number of logical units.

1.2.10 MASS MEMORY DIAGNOSTIC ROUTINES (MMDIAG)

The routine MMDIAG is included in SYSBUF and is entered from either the drum or the disk driver in the event of a mass memory failure. The error code is passed in the Q register. The alternate device handler is not called from mass memory drivers since an alternate cannot be assigned and it may be desirable to attempt recovery after printing a diagnostic message.

MMDIAG is a routine which prints a message of the following form.

MASS MEM ERR code

The error code is from 0-11. For disk, see Part II, 3.6.11; for drum, see Part II, 3.6.14

If the request which resulted in a failure was a system directory request, the routine releases the allocated core. Control then returns to the driver. Separate routines must be provided for systems with both drum and disk as MMDIAG is not re-entrant. The entry point names for these routines must be:

LABEL	OP	ADDRESS	COMMENTS
	ENT	DMDIAG	. DRUM
	ENT	DKDIAG	DISK

For disk or drum systems, remove the present EQU's which equate these entries to MMDIAG.

1.2.11 OVERLAY SUBROUTINE (OVRLAY)

The overlay subroutine, entry point OVRLAY, allows users to call for mass memory to be read over the actual call parameters. This is accomplished in the disk or drum drivers by moving the parameter list to the equipment table and using the OVRLAY subroutine to ensure that the return address from the call cannot be written over. Indirect overlay calls are not permitted. The overlay subroutine may be removed if no overlay calls are included in the system. The basic operating system, the Macro Assembler, and the FORTRAN compiler do not use the overlay subroutine.

1.2.12 PHYSICAL DEVICE TABLES (PHYSTB)

Each physical device has a PHYSTB (physical device table) which contains all device data necessary for a device to be operated by its driver. Generally this data includes:

Entry addresses to the driver responsible for operating the device

Equipment word telling the driver which device to use

Information which allows the driver to fulfill the current request

The table contains at least 13_{10} words for each device. Words 0 through 12 have a standard function for all devices. Words 13 on are used for special purposes for each driver. The system programmer should remove the device tables which are not needed for a particular system. If additional device tables are needed, use the existing device tables as a guide. However, normally make only the following changes.

The label on word 0 (ℓ)

The equipment address in word 7

Occasionally, when a driver must drive several devices, a word in the PHYSTB is used to thread one PHYSTB to another

The hardware type in bits 10 through 4 in word 8.

PHYSICAL DEVICE TABLE FORMAT (PHYSTB)

WORD	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	SYMBOLIC NAME	USE
ℓ 0	0	0	0	1	0	0	1	0	0	0	0	0					ELVL	STANDARD FOR ALL DRIVERS
1	DRIVER INITIATOR ADDRESS																EDIN	
2	DRIVER CONTINUATOR ADDRESS																EDCN	
3	DRIVER I/O HANGUP DIAGNOSTIC ADDRESS																EDPGM	
4	DIAGNOSTIC CLOCK ADDRESS																EDCLK	
5	DEVICE LOGICAL UNIT																ELU	
6	CURRENT REQUEST PARAMETER LIST ADDRESS																EPTR	
7	CONVERTER CODE					EQUIPMENT CODE				STATION CODE							EWES	
8																	EREQST	
9																	ESTAT1	
10	CURRENT BUFFER ADDRESS																ECCOR	
11	LAST WORD ADDRESS + 1 OF BUFFER																ELSTWD	
12	LAST EQUIPMENT STATUS READ																ESTAT2	
13																		USE WHEN REQUIRED BY DRIVERS OR FOR THE OUTPUT MESSAGE BUFFERING PACKAGE
14																		

Word	Bit	Significance
0		ELVL $120x A SCHDLE request to operate the driver initiator address at level x. x is the initiator priority level which should equal the priority level of the interrupt in the LOCORE program.
	14-9	Request code for SCHDLE request.
	8-4	Unused, unless specified by a particular driver.
	3-0	Priority level at which driver operates.
1		EDIN Driver initiator address (which is the second word of the SCHDLE request).
2		EDCN Driver continuator address. Control is transferred to this address (when interrupt occurred) at the priority level assigned to the interrupt in the interrupt trap region. This priority level must be the same as the priority level specified by word 0.
3		EDPGM Driver error routine address. Control is transferred to this address at the driver priority level when the diagnostic clock is counted down to negative by the diagnostic timer.
4		EDCLK Diagnostic clock. This diagnostic clock location is set by the driver and decremented by the diagnostic timer for a hardware completion interrupt. Set idle (-1) by Complete Request Routine.
5		ELU Logical unit currently assigned to device. 0 if device not in use. Set by request processor; may be reassigned by FNR routine; cleared by the next FNR routine or complete request.
6		EPTR Address of caller's parameter list. Set by FNR routine.

Word　　Bit　　Significance

7

EWES
Hardware address. To obtain equipment status: load this word into into the Q register and perform INPUT instructions. Status is saved in ESTAT2, word 12. See Control Data 1700 Computer System Code 60163500.

15　　11	10　　7	6　　0	
(W)	(E)	(S) ⟷ (D)	Q register
		command	

15-11

(W) Coverter code

Code	1706
2	#1
7	#2
C	#3
0	When coupled directly to AQ channel

10-7

(E) Equipment code. Equipment numbers for the released operating system drivers are listed in section II.4.1. Suggested equipment codes for additional drivers are in II.3.6 along with the information for each driver.

6-0

Command code. The command code is divided into two sections: S contains the station code and D contains the director function. The station code is located in bit 6 and adjacent lower order bits as required. The director function is located in bit 0 and adjacent higher order bits as required. They cannot overlap and all bits in the command code are not necessarily used. If the controller does not contain any stations, the station code is zero.

Word	Bit	Significance
8		EREQST Request status.
	15	Busy bit. 0 Operation complete 1 Operation is in progress
	14	0 If no device error 1 If driver detects device failure
	13-11	Equipment class code 0 Class not defined 1 Magnetic tape device 2 Mass storage device 3 Card device 4 Paper tape device 5 Printer device 6 Teletype device 7 Reserved for future use
	10-4	Numbers in the following list are in decimal and must be converted to hexidecimal before inserting in bits 10 through 4. Equipment type code (T). 0 1711/1712 teletypewriter 1 1721/1722 paper tape reader 2 1723/1724 paper tape punch 3 Unassigned 4 Unassigned 5 1738-853 disk unit 6 1751 drum unit 7 1729 card reader 8 1738-854 disk unit 9 601 magnetic tape unit 10 Software buffering device 11 1742 line printer 12 1728-430 card reader/punch 13 Software core allocator 14 210 CRT display station 15 1558 latching relay output 16 1553 external register output 17 311B/312B data set terminal 18 322/323 teletype terminal 19 501 line printer 20 166 line printer 21 1612 line printer 22 415 card punch 23 405 card reader 24 608 magnetic tape unit 25 609 magnetic tape unit

Word	Bit	Significance
		26 1713 teletype keyboard
		27 1713 TTY paper tape punch
		28 1713 TTY paper tape reader
		29 1729-2 card reader
		30 1797 buffered I/O interface
		31 Software dummy alternate
		32 1584 selectric I/O typewriter
		33 1582 flexowriter I/O typewriter
		34 1716 compiling data channel
		35 1718 satellite coupler
		36 Unassigned
		37 8000 series magnetic tape unit
		38 1732-608 driver
		39 1732-609 driver
		40 1530 A/D converter 30/40 PPS
		41 1534 A/D converter 200 PPS
		42 1538 A/D converter high speed
		43 Unassigned
		44 Unassigned
		45-99 Reserved for future standard equipment
		100-127 Open for user assignment
	3	0 PHYSTB does not contain message buffering in words 18-33.
		1 PHYSTB includes words 18-33 for message buffering.
	2	0 Device may not be written by unprotected programs
		1 Device may be written by unprotected programs
	1	0 Device may not be read from unprotected programs
		1 Device may be read from unprotected programs
	0	0 Device available to unprotected programs
		1 Device not available to unprotected programs
9		ESTAT1 Status word 1.
	15	0 No error occurred
		1 If error condition and/or end-of-file detected by driver
	14	0 If the number of words which were requested were transferred on a read request
		1 Set by driver if fewer words were read than requested
	13	0 No end-of-file is sensed
		1 Set by driver if device remains ready after detecting an error or end-of-file or both

Word	Bit	Significance
	12	Reserved for message interpreter request
	11	0 No error 1 Set by output message buffering package if message buffer output is incomplete
	10	0 No parity error occurred 1 Set by driver if parity error occurred
	9	Reserved
	8	Reserved for individual drivers' special use
	7	Reserved for individual drivers' special use
	6	Reserved for individual drivers' special use
	5	Data control word indicator: 0 This is not a control character 1 Set by driver if this is a control character
	4	First character of FORMAT record set by driver 0 This is not first character 1 Set by driver if this is first character
	3	Mode set 0 Set by driver when binary mode is used 1 Set by driver when ASCII mode is used
	2	Case indicator 0 Set by driver if this is lower character 1 Set by driver if this is upper character
	1	Format read/write indicator set by FNR routine 0 Unformatted record read/write 1 Formatted read or write request
	0	Read/write indicator set by FNR routine 0 Read request 1 Write request
10		ECCOR The driver will store or obtain next data from this location which was initially set by FNR routine but is updated by the driver.
11		ELSTWD The driver will satisfy the request by either storing or obtaining from this location which is the last data location +1.
12		ESTAT2 Status word 2. The last value of equipment status mentioned in word 7.
13 and beyond		Use when required by drivers or for the output message buffering package.

1.2.13 INTERRUPT RESPONSE ROUTINE

Single Device Interrupt Lines

The following example is typical of an interrupt line which serves only one device.

```
LABEL      OP         ADDRESS         COMMENTS
           ENT        I1728
I1728      LDQ        =XCD1728        PHYSTB ADDRESS

           JMP*       (CD1728+2)      GO TO CONTINUATOR
```

The addresses of the interrupt response routines must be declared as entry names, since they are externals in LOCORE.

Multiple Device Interrupt Lines

If more than one device is assigned to the interrupt line, an interrupt response routine may be coded as follows.

```
LABEL      OP         ADDRESS         COMMENTS

ATYPEI     ENA        3               NUMBER OF DEVICES-1 ON LINE
BB         STA-       I
           LDQ*       PDT, I          PHYSTB ADDRESS OF DEVICE
           LDQ-       7, Q            WORD 7 OF PHYSTB EWES
           INP        NOTIT-*         READ STATUS
           AND-       $25             $0004 MASK
           SAN        CC-*-1          SKIP IF INTERRUPT ON THIS DEVICE
NOTIT      NOP        0               NOT THIS DEVICE
           LDA-       I
           INA        -1
           SAM        ERROR-*-1       IF NO DEVICE FOUND, GO TO ERROR
           JMP*       BB              GO TO CHECK NEXT DEVICE
CC         LDQ*       PDT, I
           LDA-       2, Q
           STA-       I
DD         JMP-       (I)             GO TO CONTINUATOR
PDT        ADC        ATAB1, ATAB2,   PHYSTB ADDRESS
                      ATAB3, ATAB4
```

CAUTIONS

For some devices the status check may need to be coded differently.

Some drivers may not need a multiple device interrupt response routine. If the driver can address only one device at a time, it saves the address of the PHYSTB for the last device addressed.

Some interrupts are not associated with drivers (e.g., 1573 timer), and the interrupt response is an integral part of the program that handles the device.

1.3 SPACE

The SPACE Program includes the SPACE request processor, the allocatable core area, and the restart program. No modification is needed to the space request processor. The allocatable core area should be customized for each system.

1.3.1 ALLOCATABLE CORE

AREAC is the start of the block of allocatable core within which the mass memory resident programs are executed. The total area available is specified by the following.

```
LABEL    OP     ADDRESS

AREAC    ADC    AVCORE
         ADC    ($7FFF)
         EQU    N1($181)              Size of JOBENT        (Area 1)
         EQU    N2($4CO)              Size of JOBPRO        (Area 2)
         EQU    N3($41A)              Size of FILE          (Area 3)
         EQU    N4($1000)             Size of PROCESS       (Area 4)
         BSS    (N4-INPUT+RESTRT-1)
         BSS    AREA3(N3+2)                 Reserves the desired core
         BSS    AREA2(N2+2)                 areas and defines the starting
         BSS    AREA1(N1+2)                 address of each area
         BSS    (2)
         EQU    AVCORE(*-AREAC)
         EQU    AREA4(AREAC)
         ENT    AREA1, AREA2, AREA3, AREA4
```

The area which is now occupied by N4 can be divided into smaller areas. Example:

```
LABEL     OP       ADDRESS

          EQU      N4($21F)              Size of area 4
          EQU      N5($145)              Size of area 5
          EQU      N6($7FF)              Size of area 6
          BSS      (N6-INPUT + RESTRT-1)
          BSS      AREA5(N5+2)
          BSS      AREA4(N4+2)
          BSS      AREA3(N3+2)
          EQU      AREA6(AREAC)
          ENT      AREA5, AREA6
```

The actual definition and reservation (BSS) of the allocatable core areas is done in the SPACE Program. The table which relates these areas to each request priority is the LVLSTR table in SYSBUF.

1.3.2 RESTART PROGRAM (RESTRT)

RESTRT is the starting address of the restart program which is entered from the autoload program. The system initializer builds the autoload program during initialization and places it on the first sector (96 words) of mass memory. After transferring the image of the protected programs into core, control passes to RESTRT via location 1 in the communications region.

The MSOS 2.1 restart program also includes provision to start the 1573 timer and to schedule the diagnostic timer program. If the timer is not present or if it is switched off, a reject occurs and the message TIMER RJ is written on the comment device. The program requests the monitor to type PP and waits for the operator to acknowledge the setting of the PROGRAM PROTECT switch by typing an asterisk followed by pressing CARRIAGE RETURN. Note that a monitor request is used to type PP. If autoload does not result in PP being typed, the monitor probably was not set up properly.

Since the restart program is only used immediately after an autoload, it executes in the allocatable area, but it is set up as though it were part of the core-resident programs. In this way, the program does not require any permanent core storage, and it is destroyed as soon as a mass memory resident program is scheduled.

Modification of the restart program may be desired to allow initialization of data to occur after autoload without providing permanent core for such an initialization program. For example, code to start a process may be inserted here. Such additions may only be added just prior to the request to type PP.

1.4 MANUAL INPUT FOR PROCESS PROGRAM (MIPRO)

The manual input to the process program (MIPRO) is part of the operating system. If the input entered after a manual interrupt does not begin with an asterisk (* indicates a job processor control statement), the routine is scheduled by the manual interrupt processor (MINT) at level 3. The Q register is set to the address of the ASCII input buffer on entry to MIPRO.

If the MIPRO program is not included in the operating system at initialization time, the manual interrupt processor rejects input following a manual interrupt which does not begin with an asterisk. A J05 error message is printed.

The version of MIPRO which is supplied checks the input buffer for either DB or DX. All other inputs are rejected and the message ER is printed. If the input begins with DB, the program with the system directory entry name ODEBUG (On-Line Debug Package) is scheduled at level 3. If the input begins with DX, a flag (CHRSFG) in SYSBUF is set for the ODEBUG routine. When this flag is set, ODEBUG terminates and releases its core.

MIPRO must terminate by clearing the flag word MIB in the manual interrupt processor and then returning to the dispatcher.

MIPRO usually resides on mass storage as part of the system library, but it may be made core resident. Each user may add his own control statements to MIPRO to manually control the process.

To add a user request to MIPRO:

1. For the entry point of the request processor module, add the following with xxxxxx as the entry point

LABEL	OP	ADDRESS
	EXT	xxxxxx

2. Add to the end of the COD1 table

LABEL	OP	ADDRESS
	ALF	1, xx

xx are the same control characters used to call the user program.

3. Add the following entry in the same numeric position as it is in the COD1 table

LABEL	OP	ADDRESS
	JMP*	GETIND

4. Add the following to the INDEX table with xxxxxx as the entry point of the request processor module

LABEL	OP	ADDRESS
	ADC	(xxxxxx)

1.5 MODIFICATION OF SYSTEM FOR MINIMUM CORE REQUIREMENTS

The five modules listed in this section are included in this release for users who need a very minimum MSOS because of their system core size. These modules have not received any development activity or testing since MSOS 2.0, but they are provided for the user accustomed to them.

The five programs which differ from the corresponding module in the standard system are:

Reduced Length Modules	Size$_{10}$	Standard Modules	Size$_{10}$
MRW	86	RW	156
MMONI	51	NMONI	66
MEPROC	94	NEPROC	100
MIPROC	21	NIPROC	124
TABLES	659	LOCORE	331
	911	SYSBUF	1053
			1830

If the following restrictions are acceptable, up to 889$_{10}$ words of core can be saved by using the smaller modules instead of the standard modules.

MRW is identical to the regular version of the read/write processor (RW) except that MRW does not include ALTCHK, the routine which checks for alternate device assignments.

MIPROC causes the system to hang with no error message for parity errors.

MMONI contains in its request processor table only those requests T0 through T13. No provision for expansion is included.

TABLES combines LOCORE and SYSBUF. Locations $47 through $B2 are no longer available for process control use. The system is built to utilize only 5 interrupt lines.

The logical unit structure is as follows.

lun	Unit
1	Core allocator
2	1721 paper tape reader
3	1723 paper tape punch
4	1711 teletypewriter
5	Dummy alternate
6	Dummy
7	Dummy
8	Disk

ADDITIONAL INFORMATION ON INITIALIZATION 2

2.1 ADDITIONAL INITIALIZER CONTROL STATEMENTS

Through the use of these statements, it is possible to incorporate control statements with the actual binary programs.

2.1.1 *V ENTER STATEMENTS ON INPUT DEVICE

The *V statement instructs the system initializer to obtain subsequent control statements from the input device.

2.1.2 *U ENTER STATEMENTS ON COMMENT DEVICE

The *U statements instruct the system initializer to obtain its next and subsequent control statements from the comment device. This statement remains in effect until a *V statement is entered from the binary input device. The *U may be used to return control to the Teletypewriter wherever options may be considered (loading a special routine from another device, deleting programs, etc.).

2.1.3 *S ASSIGN ENTRY POINT NAME

*S patches external strings at system initialization time. It permits the name n to be assigned a value and to be placed in the Loader Table as an entry point. The *S statements may be used to define unpatched externals to eliminate the error printout on the listing (e.g., *S, THREE, 7FFF). The *S can also cause a program to be eliminated by doubly defining an entry point (e.g., *S, PRINT1, 7FFF). This is useful in modifying a system when the source is magnetic tape or disk.

*S, n, hhhh

This statement assigns the hexadecimal value hhhh to the entry point name n and places both in the loader table. Previously defined external strings are patched with hhhh as are future references.

*S, n, S

This statement assigns the current value of the next mass storage sector to the entry point name n. This statement permits dynamic assignment of values to symbolic names.

*S, n, P

This statement assigns the current value of the program base to the entry point name n. The program base is the next available core location into which the Initializer loads.

2.2 MESSAGES

2.2.1 SYSTEM INITIALIZER

SI	It informs the operator on the comment medium that the system initializer is ready to begin operation.
Q	Informs the operator (on comment medium) that system initializer is ready to accept another control statement.
L, nn FAILED ACTION	Appears when a driver cannot recover from an error. The operator can then take corrective action and respond with either RP or CU. RP causes the request to be repeated. CU causes the error condition to be reported to the program which made the request. Any other entry causes ACTION to be retyped.
ERROR 1	Asterisk initiator missing
ERROR 2	Number appears in name field
ERROR 3	Illegal control statement
ERROR 4	Input mode illegal
ERROR 5	No further *YM statements can be entered
ERROR 6	No further *Y statements can be entered
ERROR 7	*F statement previously entered
ERROR 8	Name appears in number field
ERROR 9	Illegal HEX core relocation field
ERROR A	Illegal mass storage sector number
ERROR B	Error return from loader module
ERROR C	Unpatched external at conclusion of *M load
ERROR D	Unpatched external at conclusion of *L load
ERROR E	Field terminator invalid
ERROR F	More than 120 characters in control statement
ERROR 10	Ordinal name without ordinal number
ERROR 11	Doubly defined entry point
ERROR 12	Invalid ordinal number
ERROR 13	*F statement not previously entered
ERROR 14	Data declared during *M load but not by first segment. Initialization restarted
ERROR 15	Attempt made to enter data into location 0 or above location $FE. Initialization restarted
ERROR 16	Irrecoverable mass storage I/O error
ERROR 17	Irrecoverable error. Last program loaded was ignored

2.2.2 PROGRAM LOADING

All loading error messages appear on the standard print output device.

E01 Irrecoverable input error; causes termination.

E03 Illegal or out-of-order input block; causes termination of load. This diagnostic also appears on the comment device when illegal input from that device is detected. The device is interrogated for a new statement.

E04 Incorrect common block storage reservation. Occurs if the largest common storage declaration is not on first NAM block to declare common storage. The loader uses the previously declared length and continues.

E05 Program too long or loader table overflow. Terminates loading. Occurs if program to be loaded exceeds available unprotected core. It may be possible to load the program by re-arranging the order of loading to insure entry points are defined before they are referred to as external symbols. Loader produces a memory map and list of unpatched externals prior to terminating the load.

E06 Attempt to load information in protected core; causes termination of load.

E07 Attempt to begin data storage beyond assigned block; causes termination of load.

E08 Duplicate entry point; loading is terminated.

E10 Unpatched external. External name is printed following diagnostic. When all unpatched externals have been printed, the operator may terminate the job or continue execution regardless of unpatched externals.

E11 Error in HEX block; loader skips remainder of block and resumes loading with the next block. The starting address is printed following diagnostic.

E12 Two programs reference same external name; one with absolute addressing, the other with relative addressing; loading is terminated.

E13 Undefined or missing transfer address; this code is not given if the loading operation is part of system initialization. Occurs when loader does not encounter a name for the transfer address or the name encountered is not defined in loader's table as entry point name.

E14 Loader request operation code word illegal.

E15 Address in I2 table is greater than $FE; issued only during system initialization. The post-resident loader initializer, I2 contains a table of information designated for locations within the communication region. An entry in this table consists of the storage address and the constant to be stored. If the address is greater than $FE, this comment is printed.

2.2.3 JOB PROCESSING

PARITY, hhhh — Memory parity error at location hhhh$_{16}$. Message appears on output comment device.

OV — Overflow of volatile storage. Message appears on output comment device.

ER — Unintelligible control statement following a manual interrupt command.

L, nn FAILED code — Informs operator of device failure.

ACTION — Requests operator action when a failed device has no alternate. The device is identified in the FAILED diagnostic.

nn — logical unit number

code — code indicating cause of failure as follows. These are typical error codes. See the individual driver for the actual error codes.

00	I/O hangup
01	Internal or external reject
02	Alarm
03	Parity error
04	Checksum error
05	Internal reject
06	External reject
07	Echo check error on punch operation
08	Illegal Hollerith punch
09	Sequence error
10	Non-negative record length
11	Change from read mode to punch mode or vice versa
12	No ${}^{7}_{9}$ punch
13	Error in disk read of mass memory driver

ALT, aa — Informs operator an alternate device, aa, has been assigned.

J01, hhhh — Program protect violation. hhhh is current contents of P register. Standard comments device.

J02, hhhh — Illegal request or parameters at location hhhh$_{16}$. Standard comment device.

J03, statement — Unintelligible control statement is output with the diagnostic. Standard comments device.

J04, statement — Illegal or unintelligible parameters in control statement. Standard comments device.

J05 — Statement entered after manual interrupt illegal. Standard comment device.

J06, hhhh — A threadable request was made at level one when no protect processor stack space was available, or an unprotected threaded request was made at level one. Standard comments device.

J07, hhhh — Unprotected program tried to access protected device from location hhhh. Standard comments device.

J08, hhhh — Attempt to access read only unit for write, or write only unit for read, or an attempt to access an unprotected request on a protected unit. Standard comments device.

2.2.4 DEBUGGING AND LIBRARY EDITING

The following messages appear on the output comment device. Both the system initializer and LIBEDT will attempt error recovery whenever possible. Illegal input statements are not processed.

BP, hhhh	Breakpoint program ready for input. The breakpoint address hhhh$_{16}$ is printed only if breakpoint program was entered from previously set breakpoint.
B01, statement	Statement or parameters are unintelligible for the breakpoint program.
B02, hhhh	hhhh$_{16}$ cannot be processed by breakpoint program because it is protected.
B03, hhhh	Breakpoint limit exceeded. hhhh$_{16}$ is the last breakpoint processed.
B04	Previous *E statement requested entries in protected core. Entries are not processed; breakpoint program waits for new statement.
RE	Recovery program ready to accept control statements.
LIB	Library editing program ready to accept control statements.
J	Job processor waiting for control statement from input comment device.
L01	More than six characters in a parameter name presented to the library editing program.
L02	More than 6 digits in a number presented to the library editing program.
L03	Improper system directory ordinal presented to the library editing program.
L04	Invalid control statement presented to the library editing program.
L05	Illegal field delimiter in a control statement presented to the library editing program.
L06	Illegal field in control statement presented to the library editing program.
L07	Errors in loading as a result of a library editing program control statement.
L08	A program to be added to the program library has an entry point duplicating one already in the directory.
L09	Standard input failed on first input record following an *N request.
L10	The operator is deleting a program which is not in the library.
L11	No header record on file input from mass storage.
L12	On an * L, entry statement, either there was an input error, or the first record was not a NAM block.
L13	Common declared by the program being loaded exceeds available common.
L14	Program being loaded is longer than the size of unprotected core, but not longer than the distance from the start of unprotected core to the top of core.

INDEX

INDEX

CONTROL DATA
CORPORATION

COMMENT AND EVALUATION SHEET

1700 MSOS INSTALLATION HANDBOOK

Pub. No. 60234300E August 1970

THIS FORM IS NOT INTENDED TO BE USED AS AN ORDER BLANK. YOUR EVALUATION OF THIS MANUAL WILL BE WELCOMED BY CONTROL DATA CORPORATION. ANY ERRORS, SUGGESTED ADDITIONS OR DELETIONS, OR GENERAL COMMENTS MAY BE MADE BELOW. PLEASE INCLUDE PAGE NUMBER REFERENCE.

FROM NAME: ____________________

BUSINESS
ADDRESS: ____________________

____

NO POSTAGE STAMP NECESSARY IF MAILED IN U. S. A.

FOLD ON DOTTED LINES AND STAPLE

FOLD FOLD

FIRST CLASS
PERMIT NO. 8241

MINNEAPOLIS, MINN.

BUSINESS REPLY MAIL

NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A.

POSTAGE WILL BE PAID BY

CONTROL DATA CORPORATION
Software Documentation
4201 North Lexington Avenue
St. Paul, Minnesota 55112

FOLD FOLD

STAPLE STAPLE